Computer Science Department

TECHNICAL REPORT

Random B-trees with Inserts and Deletes

T. Johnson
D. Shasha

Technical Report 453

June 1989

NEW YORK UNIVERSITY

Department of Computer Science
Courant Institute of Mathematical Sciences
251 MERCER STREET, NEW YORK, N.Y. 10012

Random B-trees with Inserts and Deletes

T. Johnson
D. Shasha

Technical Report 453

June 1989

Random B-trees with Inserts and Deletes

Theodore Johnson, johnsnth@csd2.nyu.edu
Dennis Shasha, shasha@nyu.edu
Courant Institute of Mathematical Sciences, New York University[*]

June 10, 1989

Abstract

The utilization of B-tree nodes determines the number of levels in the B-tree and hence its performance. Yao showed that the utilization of B-tree nodes under pure inserts was 69%. We derive analytically and verify by simulation the utilization of B-tree nodes constructed from a mixture of insert and delete operations. Assuming that nodes only merge when they are empty we show that the utilization is 39% when the number of inserts is the same as the number of deletes. However, if there are just 5% more inserts than deletes, then the utilization is over 62%. We also calculate the probability of splitting and merging. We derive a simple rule-of-thumb that accurately calculates the probability of splitting. We also model B-trees that merge half-empty nodes. The utilization of merge-at-half B-trees is slightly larger than the utilization of merge-at-empty B-trees (within 10% if if there are at least 5% more inserts than deletes), but the restructuring rate is much higher. We present two models for computing B-tree utilization, the more accurate of which remembers items inserted and then deleted in a node. After analyzing the leaf level of a random B-tree, we analyze the upper levels, and show that they have the structure of the leaf level of a pure-insert tree, independent of the percentage of operations that are deletes (if the percentage of deletes is less than 50%).

1 Introduction

B-trees are commonly used in very large database systems to provide indices into the database. The space that the B-tree index consumes can be considerable. In order to allocate an appropriate amount of space for the B-tree, an estimate of the B-tree utilization is needed.

In this paper, we will derive and solve equations that describe the equilibrium structure of a B-tree under a parameterized mix of insert and delete operations, where the deletes can come from modify operations. Every item in the B-tree is equally likely to be deleted on a delete operation. Every permutation of the operands of the insert operations is equally likely. The method is based on Yao's [Yao 78] except that we will consider the solution to be the equilibrium point of a set of difference equations. We will show how the probability that a node receives an insert can be calculated by keeping track of the number of *ghosts*, or items that have been deleted, that are in the B-tree.

In addition to the utilization, we will calculate the probability of splitting or merging a node on the bottom level. After analyzing the leaf level of a random B-tree, we will analyze the upper levels of a B-tree. We will use this analysis to predict the height and number of children of the root.

We will model two types of B-trees in this paper. The first type allows B-tree nodes to become arbitrarily small (i.e. contain as few as one entry). When a node becomes empty, the space is freed and the delete is propagated upwards. We call this type of B-tree a *merge-at-empty* B-tree. Merge-at-empty

[*]This work was partially supported by the National Science Foundation under grant number DCR8501611 and by the Office of Naval Research under grant number N00014-85-K-0046.

B-trees are commonly used in database management systems. The second type of B-tree that we will model merges a half-empty node with its neighbor on a delete (a *merge-at-half* B-tree). We will compare the two types of B-trees in this paper and conclude that database management systems have made the right decision.

1.1 Simulation Model

We wrote a B-tree simulator to compare against our analytical results. The simulation builds a B-tree out of a sequence of inserts and deletes, then applies a long sequence of parameterized inserts and deletes. Every item in the B-tree is equally likely to be chosen as the operand of a delete operation. The operands of the insert operations are chosen uniformly randomly from a large key-space.

1.2 Previous Work

Yao derived an estimate of 69% utilization for pure insert operations in [Yao 78]. A paper that independently derived the same result by nearly the same methods is [Nakamura 78]. Yao's method of analysis is generalized to *fringe* analysis in [Eisenbarth 82]. Fringe analysis is the analysis of the leaves of a tree data structure. Eisenbarth et.al. showed how to solve the matrix recurrence equations that result from a pure-insert model. B-tree variants were analyzed using fringe analysis in [Baeza-Yates 89a] and [Baeza-Yates 89b]. Another B-tree variant that has high space utilization was analyzed in [Kuspert 83]. An alternative approach to estimating the utilization of a B-tree appeared in [Leung 84] and was improved on in [Gupta 86]. Yao's method was elaborated upon to obtain the probability of splitting in [Wright 85]. The number of children of the root of a B-tree was approximated in [Driscoll 87].

None of the above papers addressed the problem of deletes in the instruction mix. Mizoguchi [Mizoguchi 79] proposed an approximate model for merge-at-empty B-trees in order to analyze utilization. The range of his analysis was from pure inserts to 33% deletes/77% inserts. He also predicted the utilization at 50% deletes/50% inserts (pure modifies), but his solution was pessimistic. Our linear model is similar to Mizoguchi's. (We show that the linear model is not a good approximation as the percentage of deletes approaches 50%.)

Quitzow and Klopprogge [Quitzow 80] proposed a differential equation model to predict the utilization for both merge-at-empty and merge-at-half B trees. They analyzed the case of pure inserts (where they are consistent with Yao) and the case of pure modifies. Our linear model is similar to theirs.

Wright and Langenhop [Langenhop 85] also have a model for merge-at-half B trees with pure modify operations. Their predictions are different from those in [Quitzow 80] (and ours). It turns out that simulation results fall roughly in between.

Hsu and Zhang [Zhang 87] analyzed restructuring rates (i.e. rates of splits and merges) as a function of the merging point using a simple mathematical model. Our qualitative conclusions are similar to theirs, but our quantitative predictions are significantly different.

So, the main contributions of the present work are to show the limits of the linear model (as compared with the ghost model), an analysis of the utilization as the number of deletes approaches the number of inserts (indicating the sharp knee in the utilization curve), a set of predictions of restructuring rates as characterized by a simple rule of thumb, and a comparison of the analytical models with a simulation of an actual B tree.

2 Merge-at-Empty Btrees

2.1 Pure Inserts

In this section, we review Yao's analysis for pure inserts, because our analysis is a generalization of Yao's. We begin with the terminology and methodology.

A *B-tree* is a balanced tree in which the distance between the root and any leaf is the same (Knuth 73]). A B-tree is of *parameter p* if each non-leaf node has between 1 and $2p-1$ children. The children of an interior node are accounted for by *entries* in a node; there is one entry per child. The entries contain key and pointer information. The leaf nodes contain the *items* in the B-tree. A B-tree node is of *order* k if it has k entries.

Define:

N: number of inserts (assumed successful).

$X_i(N)$: number of order i nodes in a particular B-tree of size N (i.e. resulting from N inserts).

$A_i(N)$: expected number of order i nodes in a random B-tree of size N.

$a_i(N)$: $= A_i(N)/N$.

U: expected space utilization of a random B-tree

P_s: probability of splitting on an insert

Let $t[N]$ be a particular random B-tree of parameter p with N items. Then $X_i^{t[N]}$ is the number of leaf-level nodes in $t[N]$ of order i. Let $\Pr[t[N]]$ be the probability that a random B-tree of parameter p with N items is $t[N]$. Define $A_i(N)$ to be the expected number of leaf-level nodes of order i in a tree with N items. Then

$$A_i(N) = \sum_{t[N]} X_i^{t[N]} \Pr[t[N]]$$

A B-tree of parameter p with N items is of *class* $(X_1, X_2, \ldots, X_{2p}|N)$ if it has X_1 nodes of order 1 on the leaf level, X_2 nodes of order 2 on the leaf level, etc. Let $\Pr[(X_1, \ldots, X_{2p-1}|N)]$ be the probability that a random B-tree of parameter p with N items is of class $(X_1, \ldots, X_{2p-1}|N)$. Another way to calculate A_i is:

$$A_i(N) = \sum_{(X_1,\ldots,X_i,\ldots,X_{2p-1}|N)} X_i \Pr[(X_1, \ldots, X_i, \ldots, X_{2p-1}|N)]$$

Suppose that we have a B-tree of class $(X_p, \ldots, X_{2p-1}|N)$[1] and an insertion occurs. If the insertion goes to an order i node, $p \le i \le 2p-1$, then the result is a B-tree of class $(X_p, \ldots, X_i - 1, X_{i+1} + 1, \ldots, X_{2p-1}|N+1)$. If the insertion goes to an order $2p-1$ leaf node, then the result is a B-tree of class $(X_p + 2, \ldots, X_{2p-1} - 1|N+1)$, because splitting an order $2p-1$ node results in two order p nodes. Next, in order to specify completely the way that a B-tree evolves through inserts, we must specify the probability that an insert is directed to an order i node.

To do that, specify an input probability distribution. We will use a uniform distribution in the following sense: Any of the $N!$ permutations of an N item insert sequence are equally likely. If N items have already been inserted, then there are $N+1$ positions to which the next insert can be directed to ($N-1$ intervals and 2 end positions), each of which is equally likely. The probability that a given node of order i receives an insert is thus $i/(N+1)$. In addition if the node is, say, the rightmost one, then the node has an additional probability of $1/(N+1)$ of receiving the insert. Therefore, the probability that an insert is directed to an order i node in a class $(X_p, \ldots, X_i, \ldots, X_{2p-1}|N)$ B-tree is

$$\frac{iX_i}{N+1} + \frac{1}{N+1}\Pr[\text{the rightmost item is in an order i node}]$$

$$= \frac{iX_i}{N+1} + \frac{1}{N+1}\frac{iX_i}{N}$$

[1] $X_1 = X_2 = \cdots = X_{p-1} = 0$

$$= \frac{iX_i}{N}$$

Using the above probabilities, we can develop a system of recurrence equations for the $A_i(N)$

$$\begin{aligned} A_p(N+1) = & \sum_{(X_p,\ldots,X_{2p-1}|N)} \Pr[(X_p,\ldots,X_{2p-1}|N)]\Big\{\tfrac{pX_p}{N}(X_p-1)+ \\ & \tfrac{(2p-1)X_{2p-1}}{N}(X_p+2)+\tfrac{N-pX_p-(2p-1)X_{2p-1}}{N}X_p\Big\} \\ A_i(N+1) = & \sum_{(X_p,\ldots,X_{2p-1}|N)} \Pr[(X_p,\ldots,X_{2p-1}|N)]\Big\{\tfrac{iX_i}{N}(X_i-1)+ \\ & \tfrac{(i-1)X_{i-1}}{N}(X_i+1)+\tfrac{N-iX_i-(i-1)X_{i-1}}{N}X_i\Big\} \qquad i \neq p \end{aligned}$$

These reduce to

$$A_p(N+1) = A_p(N)(1-p/N) + A_{2p-1}(N)\frac{2(2p-1)}{N}$$

$$A_i(N+1) = A_i(N)(1-i/N) + A_{i-1}(N)\frac{i-1}{N}$$

Making the substitution $a_i(N) = A_i(N)/N$

$$(N+1)a_p(N+1) = a_p(N)(N-p) + 2(2p-1)a_{2p-1}(N)$$

$$(N+1)a_i(N+1) = a_i(N)(N-i) + (i-1)a_{i-1}(N)$$

The $a_i(N)$ describe the proportion of nodes on the leaf level that are of order i. Up to this point, we have been following Yao's treatment of the pure-insert problem. Next, however, we will deviate from Yao's methods and transform the equations into a set involving terms of the form $\Delta a_i(N+1) = a_i(N+1) - a_i(N)$:

$$\begin{aligned} (N+1)\Delta a_p(N+1) = & \; -a_p(N)(p+1) + 2(2p-1)a_{2p-1}(N) \\ (N+1)\Delta a_i(N+1) = & \; -a_i(N)(i+1) + (i-1)a_{i-1}(N) \qquad i \neq p \end{aligned}$$

We want to solve for the equilibrium point of this set of simultaneous difference equations – that is, the point where $\Delta a_i(N+1) = 0$. At the equilibrium point, the $a_i(N)$ will not change with N, so we can remove the dependence on N to obtain the set of equations:

$$\begin{aligned} 0 = & \; -(p+1)a_p + 2(2p-1)a_{2p-1} \\ 0 = & \; -(i+1)a_i + (i-1)a_{i-1} \qquad i \neq p \end{aligned}$$

In order to prove the existence and uniqueness of the equilibrium point, make the substitution $P_i = ia_i$. Then we have the equations

$$\begin{aligned} 0 = & \; -(p+1)P_p + 2pP_{2p-1} \\ 0 = & \; -(i+1)P_i + iP_{i-1} \qquad i \neq p \end{aligned}$$

The above equations sum to 0, so it is easy to see that the above p equations are of rank $p-1$. Therefore, the matrix of coefficients is singular, so that the set of equations has a non-trivial solution. That the matrix is of rank $p-1$ also means that it has only one eigenvalue of 0, so the equilibrium point is unique. Since the equilibrium point of the P_i exists and is unique, the equilibrium point of the a_i exists and is unique also. The solution that we are looking for is subject to

$$\sum_{i=p}^{2p-1} ia_i = 1.$$

The stability of the solution follows from the fact that non-equilibrium states will tend towards equilibrium states, as can be seen by considering the system to be a system of flows. Think of each a_i as being a cell. The flow out of a cell is directly proportional to the value of the cell, the flow in is directly proportional to the value of the neighbors. Suppose that a cell has less than its equilibrium value. Because $\sum_{i=p}^{2p-1} ia_i = 1$, some other cells must have more than their equilibrium value. Therefore the flow out of the cell will be less than at equilibrium, and the flow in will be larger than at equilibrium. The result is a net flow in, and the value of the cell will grow. The converse holds if a cell has greater than its equilibrium value. This argument generalizes to a sequence of cells with less or more than their equilibrium values.

We have a system of simultaneous linear equations that we can solve numerically. However, this system is simple enough that we can give an algebraic solution.

Theorem 1 ([Yao 78])

$$a_i = \frac{1}{i(i+1)}(H(2p) - H(p))^{-1} \quad p \le i \le 2p-1$$

Where $H(p)$ is the harmonic function $H(p) = \sum_{i=1}^{p} 1/i \approx \ln p$.

The following lemma has appeared in [Wright 85]:

Lemma 1 *The probability of inserting at an order i node on the leaf level when all of the operations on the B-tree are inserts is ia_i.*

Corollary 1 *The probability of inserting at a full node on the leaf level when all of the operations on the B-tree are inserts is*

$$P_s = (2p-1)a_{2p-1}$$

The value $(2p-1)a_{2p-1}$ is approximately $1/(2p\ln 2)$ when p is large (where the maximum node size is $2p-1$). This contradicts the false intuition that the probability of splitting at the leaf level would be $1/p$, as p insertions into an half-empty node causes a split. The reason is that the B-tree is growing; the factor of $2\ln 2$ in the denominator is the result.

The *space utilization*, U, of a B-tree is the portion of space taken up by the B-tree that store information. The following lemma, which has appeared in [Mizoguchi 79] and [Quitzow 80], will also be useful:

Lemma 2 *If U is the utilization of the B-tree, then*

$$U = \frac{1}{(2p-1)\sum a_j}$$

Corollary 2 *The utilization of a B-tree with pure-insert operations is*

$$U = \frac{2p[H(2p) - H(p)]}{2p-1} \approx 69\%$$

Where $H(p)$ is the harmonic function defined previously.

Proof: Sum the a_i and apply the lemma. ♣

2.2 Pure-Modify Operations

The next instruction mix that we will examine is *pure-modify*. Every operation is assumed to be a modify operation, which is modeled as a delete followed by an insert. That is, a modify deletes one key then inserts one key. We assume that the B-tree is initially built from some sequence of operations (which may contain some deletes), then has a long sequence of modify operations applied to it. Both the insert and delete of the modify operation are assumed to be successful.

2.2.1 Ghost Model

Both [Mizoguchi 79] and [Quitzow 80] have analyzed the case of pure-modify operations on merge-at-empty B-trees, but they assume that the probability that a node receives an insert is proportional to its size. The problem with this model is that after a large number of modify operations, the number of items in a node bears little relation to the number of insert operations that were performed on the node, yet the probability that a node will receive an insert is related to the number of inserts performed on the node. We will define a *ghost* to be a data item that was inserted, then later deleted. Though the ghost no longer exists in the B-tree, it still affects the distribution of insert operations. We must therefore keep track of the number of ghosts at each node.

Define:

N: the number of items in the B-tree.

M: number of modify operations.

$A_i(M)$: expected number of order i nodes after modify operation M.

$B_i(M)$: expected number of order i nodes after the delete portion of modify operation M.

$C_i(M)$: the average number of ghosts contained in all of the order i nodes after M modify operations.

$C_i'(M)$: The average number of ghosts contained in an order i node after M modify operations. $C_i'(M) = C_i(M)/A_i$.

$c_i(M)$: The proportion of ghosts that are in order i nodes after M modifies. $c_i(M) = C_i(M)/M$.

$D_i(M)$: the average number of ghosts contained in all of the order i nodes after the delete portion of modify operation M.

$D_i'(M)$: The average number of ghosts contained in an order i node after the delete portion of modify operation M. $C_i'(M) = C_i(M)/A_i$.

$d_i(M)$: The proportion of ghosts that are in order i nodes after the delete portion of modify operation M. $c_i(M) = C_i(M)/M$.

v: $= \sum a_i$.

P_m: probability of removing (merging) a node because of a delete.

Let us first develop the equations that describe the a_i. After the delete portion of the modify:

$$B_i(M+1) = (1 - i/N)A_i(M) + \frac{i+1}{N}A_{i+1}(M) \quad i \neq 2p-1$$
$$B_{2p-1}(M+1) = (1 - \frac{2p-1}{N})A_{2p-1}(M)$$

Next we turn to the insert. The number of nodes of order i after an insert is the number of nodes before the insert, minus the expected loss, plus the expected gain:

$$A_i(M+1) = (\mathit{previous}) - (\Pr[\text{order i node hit}])(\mathit{loss}) - (\Pr[\text{order i} - 1 \text{ node hit}])(\mathit{gain})$$

Here, the previous value is $B_i(M+1)$ and the gain or loss is 1. After a large number of modifies, there will be many more ghosts than data items in the tree. Therefore, the probability that a node of order i

receives the insert approaches $c_i(M)$.

$$\begin{aligned}
A_i(M+1) = {} & B_i(M+1) - c_i(M)\cdot 1 + c_{i-1}(M)\cdot 1 \\
& \text{substitute in the value of } B_i(M): \\
= {} & (1-i/N)A_i(M) + \tfrac{i+1}{N}A_{i+1}(M) - c_i(M) - c_{i-1}(M) \\
a_i(M+1) = {} & (1-i/N)a_i(M) + \tfrac{i+1}{N}a_{i+1}(M) - c_i(M)/N + c_{i-1}(M)/N
\end{aligned}$$

Solving for $\Delta a_i(M+1)$,

$$N\Delta a_i(M+1) = \ -ia_i(M) + (i+1)a_{i+1}(M) - c_i(M) + c_{i-1}(M)$$

The equations for the $a_i(M)$ depend on values for the $c_i(M)$. We next develop equations that describe the $c_i(M)$. Again model a modify as a delete followed by an insert, and calculate the new number of ghosts as the previous number minus the expected loss plus the expected gain. Start with the delete:

$$D_i(M+1) = (\mathit{previous}) - (\Pr[\text{order i node hit}])(\mathit{loss}) + (\Pr[\text{other node hit}])(\mathit{gain})$$

The previous value is $C_i(M)$. The probability that a delete is directed to an order i node is $iA_i(M)/N$, the proportion of data items contained in order i nodes. The loss is $C_i'(M)$, the expected number of ghosts in an order i node. D_i will gain if a delete was directed to an order $i+1$ node or to an order 1 node. The gain from an order $i+1$ node is $C_{i+1}'(M)+1$, because a new ghost is created. If a delete is directed to an order 1 node, then the node is merged. The node immediately to the right will receive the merged node's key range - thus will receive the merged node's ghosts. The probability that an order i node receives the deleted node's ghosts is $a_i(M)/\sum_{j=1}^{2p-1} a_j(M)$. Therefore:

$$\begin{aligned}
D_i(M+1) = {} & C_i(M) - (\tfrac{iA_i(M)}{N})(C_i'(M)) + (\tfrac{(i+1)A_{i+1}(M)}{N})(C_{i+1}'(M)+1) \\
& +(\tfrac{A_1(M)}{N})(\tfrac{a_i(M)}{\sum_{j=1}^{2p-1} a_j(M)})(C_1'(M+1)+1) \\
= {} & C_i(M) - (\tfrac{iA_i(M)}{N})(\tfrac{C_i(M)}{A_i(M)} + (\tfrac{(i+1)A_{i+1}}{N})(\tfrac{C_{i+1}(M)}{A_{i+1}}+1) \\
& +(\tfrac{A_1(M)}{N})(\tfrac{a_i(M)}{\sum_{j=1}^{2p-1} a_j(M)})(\tfrac{C_1'(M+1)}{A_1(M)}+1) \\
= {} & C_i(M) + \tfrac{iC_i(M)}{N} + \tfrac{(i+1)C_{i+1}(M)}{N} + \tfrac{a_i(M)C_1(M)}{N\sum_{j=1}^{2p-1} a_j(M)} + (i+1)a_{i+1}(M) + \tfrac{a_i(M)a_1(M)}{\sum_{j=1}^{2p-1} a_j(M)} \\
\frac{D_i(M+1)}{M} = {} & c_i(M) + \tfrac{ic_i(M)}{N} + \tfrac{(i+1)c_{i+1}(M)}{N} + \tfrac{a_i(M)c_1(M)}{N\sum_{j=1}^{2p-1} a_j(M)} + \tfrac{(i+1)a_{i+1}(M)}{M} + \tfrac{a_i(M)a_1(M)}{M\sum_{j=1}^{2p-1} a_j(M)}
\end{aligned}$$

The terms $((i+1)a_{i+1}(M))/M$ and $(a_i(M)a_1(M))/(M\sum_{j=1}^{2p-1} a_j(M))$ become insignificant as M becomes large, so we may drop them. Since $D_i(M+1)/M = D_i(M+1)/(M+1) + D_i(M+1)/[M(M+1)]$, we may assume that $D_i(M+1)/M \approx d_i(M+1)$, so that:

$$\begin{aligned}
d_i(M+1) = {} & c_i(M) + \tfrac{ic_i(M)}{N} + \tfrac{(i+1)c_{i+1}(M)}{N} + \tfrac{a_i(M)c_1(M)}{N\sum_{j=1}^{2p-1} a_j(M)} \quad & i = 1,\ldots,2p-2 \\
d_i(M+1) = {} & c_i(M) + \tfrac{ic_i(M)}{N} + \tfrac{a_i(M)c_1(M)}{N\sum_{j=1}^{2p-1} a_j(M)} \quad & i = 2p-1
\end{aligned}$$

The $C_i(M+1)$ evolve from the $D_i(M+1)$ in the following way:

$$C_i(M+1) = (\mathit{previous}) - (\Pr[\text{order i node hit}])(\mathit{loss}) + (\Pr[\text{order i} - 1 \text{ node hit}])(\mathit{gain})$$

The previous is $D_i(M+1)$. The probability that an insert is directed to an order i node is $d_i(M+1)$, and the gain or loss is $D_i'(M+1)$. Therefore:

$$\begin{aligned}
C_i(M+1) &= D_i(M+1) - (d_i(M+1))(D_i(M+1)/A_i(M)) + (d_{i-1}(M+1))(D_{i-1}(M+1)/A_{i-1}(M)) \\
c_i(M+1) &= d_i(M+1) - \frac{d_i(M+1)^2}{A_i(M)} + \frac{d_{i-1}(M+1)^2}{A_{i-1}(M)} \\
c_i(M+1) &= c_i(M) - \frac{ic_i}{N} + \frac{(i+1)c_{i+1}(M)}{N} + \frac{a_i(M)c_1(M)}{N\sum_{j=1}^{2p-1} a_j(M)} - \frac{d_i(M+1)^2}{A_i(M)} + \frac{d_{i-1}(M+1)^2}{A_{i-1}(M)}
\end{aligned}$$

At the equilibrium point, the $a_i(M)$, $c_i(M)$ and the $d_i(M)$ are constant, so remove their dependence on M Including the exceptional cases at $i = 1, p, 2p-1$, the equations that determine the equilibrium point of the a_i are:

$$\begin{array}{rcll}
0 = & -a_i + (i+1)a_{i+1} - c_i & i = 1 & (1.1) \\
0 = & -ia_i + (i+1)a_{i+1} - c_i + c_{i-1} + 2c_{2p-1} & i = p & (1.2) \\
0 = & -ia_i - c_i + c_{i-1} & i = 2p-1 & (1.3) \\
0 = & -ia_i + (i+1)a_{i+1} - c_i + c_{i-1} & \text{otherwise} & (1.4)
\end{array}$$

In addition, $\sum_{i=1}^{2p-1} ia_i = 1$. If we solve for the point at which $N\Delta c_i(M+1) = 0$, the following equations describe the equilibrium point of the c_i:

$$\begin{array}{rcl}
0 = & -ic_i + (i+1)c_{i+1} + \frac{a_i c_1}{\sum_{j=1}^{2p-1} a_j} - \frac{d_i^2}{a_i} & i = 1 \\
0 = & -ic_i + (i+1)c_{i+1} + \frac{a_i c_1}{\sum_{j=1}^{2p-1} a_j} - \frac{d_i^2}{a_i} + \frac{d_{i-1}^2}{a_{i-1}} + \frac{d_{2p-1}^2}{a_{2p-1}} & i = p \\
0 = & -ic_i + \frac{a_i c_1}{\sum_{j=1}^{2p-1} a_j} - \frac{d_i^2}{a_i} + \frac{d_{i-1}^2}{a_{i-1}} & i = 2p-1 \\
0 = & -ic_i + (i+1)c_{i+1} + \frac{a_i c_1}{\sum_{j=1}^{2p-1} a_j} - \frac{d_i^2}{a_i} + \frac{d_{i-1}^2}{a_{i-1}} & \text{otherwise}
\end{array}$$

Additionally, $\sum_{i=1}^{2p-1} c_i = 1$.

We can simplify the equations beyond this: Consider equations (1.1 - 1.4). Each is parameterized for a certain set of values of i. Let the equation for a particular i be denoted by e_i. Add e_1 and e_2 to get:

$$0 = 3a_3 - c_2 - a_1$$

Adding e_3 to this, we get

$$0 = 4a_4 - c_2 - a_1$$

This form continues up to e_{p-1}

$$0 = pa_p - c_{p-1} - a_1$$

Adding e_p, we get

$$(p+1)a_{p+1} - c_p + 2c_{2p-1} - a_1$$

This form continues until e_{2p-1}

$$(2p-1)a_{2p-1} - c_{2p-2} + 2c_{2p-1} - a_1$$

Adding e_{2p-1}, we get:

$$a_1 = c_{2p-1}$$

Substituting c_{2p-1} for a_1 in the previous equations, we get:

$$\begin{array}{rll} a_1 = c_{2p-1} & & \\ ia_i = c_{i-1} + c_{2p-1} & & 2 \le i \le p \\ ia_i = c_{i-1} - c_{2p-1} & & p+1 \le i \le 2p-1 \end{array}$$

Examining the equations for the d_i, we see that $d_i \to c_i$ as $N \to \infty$. Let:

$$v = \sum_{i=1}^{2p-1} a_i = \sum_{i=1}^{2p-1} c_i/(i+1) + [2H(p) - H(2p-1)]c_{2p-1}$$

Using this and the equations for the a_i, we get:

Theorem 2 *The values of c_i, $1 \le i \le 2p-1$ satisfy the following system of non-linear equations:*

$$\begin{array}{lcl}
0 = & -c_1 + 2c_2 + \frac{c_{2p-1}c_1}{v} - \frac{c_1^2}{c_{2p-1}} & i = 1 \\
0 = & -2c_2 + 3c_3 + \frac{(c_1+c_{2p-1})c_1}{2v} - \frac{2c_2^2}{(c_1+c_{2p-1})} + \frac{c_1^2}{c_{2p-1}} & i = 2 \\
0 = & -ic_i + (i+1)c_{i+1} + \frac{(c_{i-1}-c_{2p-1})c_1}{iv} - \frac{ic_i^2}{c_{i-1}+c_{2p-1}} + \frac{(i-1)c_{i-1}^2}{c_{i-1}+c_{2p-1}} & 2 < i < p \\
0 = & -pc_p + (p+1)c_{p+1} + \frac{(c_{p-1}+c_{2p-1})c_1}{pv} - \frac{pc_p^2}{c_{p-1}+c_{2p-1}} + \frac{(p-1)c_{p-1}^2}{c_{p-1}+c_{2p-1}} + \frac{(2p-1)c_{2p-1}^2}{c_{2p-2}-c_{2p-1}} & i = p \\
0 = & -ic_i + (i+1)c_{i+1} + \frac{(c_{i-1}-c_{2p-1})c_1}{iv} - \frac{ic_i^2}{c_{i-1}-c_{2p-1}} + \frac{(i-1)c_{i-1}^2}{c_{i-2}+c_{2p-1}} & i = p+1 \\
0 = & -ic_i + (i+1)c_{i+1} + \frac{(c_{i-1}-c_{2p-1})c_1}{iv} - \frac{ic_i^2}{c_{i-1}-c_{2p-1}} + \frac{(i-1)c_{i-1}^2}{c_{i-1}-c_{2p-1}} & p+1 < i < 2p-1 \\
0 = & -(2p-1)c_{2p-1} + \frac{(c_{2p-2}-c_{2p-1})c_1}{(2p-1)v} - \frac{(2p-1)c_{2p-1}^2}{c_{2p-2}-c_{2p-1}} + \frac{(2p-2)c_{2p-2}^2}{c_{2p-3}-c_{2p-1}} & i = 2p-1 \\
0 = & 1 - \sum_{j=1}^{2p-1} c_j &
\end{array}$$

The equations for the Δc_i are linearly dependent: they sum to 0. Remove one of the equations to get $2p-1$ equations in $2p-1$ variables, and the system is ready to be solved by a non-linear equation solving package. The package we used is NAG [NAG].

Three simulation experiments were run. A B-tree was built using insert operations, and then a long sequence of modify operations was performed. The experiments were run for $p = 5, 10, 15, 20$. Table 2 compares the utilization, U, and probability of splitting, P_s, from the simulation and the ghost model. The calculated values for the utilization are within 10% of the utilization observed in the simulation. The calculated and observed probabilities of splitting diverge, but they both decrease exponentially with p. Figure 1 compares the distribution of the a_i. Figure 2 shows the calculated utilization for p between 5 and 25. In the model as well as the simulation, the utilization approaches 39%, and the probability of splitting or deleting a node decreases exponentially with p.

2.3 Parameterized Inserts and Deletes

An intermediate operation mix between pure-insert and pure-modify is the case where there are deletes in the operation mix, but there are more inserts than deletes. We will call this operation mix *parameterized by q*. Let us define:

L: number of operations performed.

q: probability that a given operation is a delete.

Therefore, an insert will occur with probability $1-q$ and the expected number of items in the tree after L operations will be $L(1-2q)$.

2.3.1 Ghost Model

Because there are deletes in the instruction mix, but there are more inserts than deletes, both the number of items and the number of ghosts in the leaves will grow with L. The expected number of items in the tree will be $L(1-2q)$ after L operations, and the expected number of ghosts will be Lq, so that the total expected number of ghosts and items will be $L(1-q)$. Therefore $C_i = Lqc_i$ and $A_i = L(1-2q)a_i$. The actual number of items and ghosts in the B-tree will have a binomial distribution, but by the law of large numbers, the deviation will become negligible compared to the mean.

The equations that describe the A_i, $1 < i < 2p-1$ and $i \neq 1$ are:

$$\begin{aligned}
A_i(L+1) = {} & q\left(\tfrac{-i}{L(1-2q)}A_i(L) + \tfrac{i+1}{L(1-2q)}A_{i+1}(L)\right) + \\
& (1-q)\left(\tfrac{C_i(L)+iA_i(L)}{L(1-q)}(-1) + \tfrac{C_{i-1}(L)+(i1-)A_{i-1}(L)}{L(1-q)}\right) + A_i(L) \\
(L+1)(1-2q)a_i(L+1) = {} & -iqa_i(L) + (i+1)qa_{i+1}(L) - qc_i(L) - i(1-2q)a_i(L) \\
& +qc_{i-1}(L) + (i-1)(1-2q)a_{i-1}(L) + L(1-2q)a_i(L) \\
(L+1)(1-2q)\Delta a_i(L+1) = {} & -(i(1-q)+(1-2q))a_i - qc_i + (i+1)qa_{i+1} + qc_{i-1} + (i-1)(1-2q)a_{i-1}
\end{aligned}$$

The equations that describe the C_i, $1 < i < 2p-1$ and $i \neq 1$ are:

$$\begin{aligned}
C_i(L+1) = {} & q\left[\tfrac{iA_i(L)}{L(1-2q)}\left(-\tfrac{C_i(L)}{A_i(L)}\right) + \tfrac{(i+1)A_{i+1}}{l(1-2q)}\left(\tfrac{C_{i+1}(L)}{A_{i+1}(L)}+1\right) + \left(\tfrac{A_1(L)}{L(1-2q)}\right)\left(\tfrac{A_i(L)}{\sum A_j(L)}\right)\left(\tfrac{A_1(L)}{C_1(L)}+1\right)\right] \\
& +(1-q)\left(\tfrac{C_i(L)+iA_i(L)}{L(1-q)}\left(\tfrac{C_i(L)}{A_i(L)}\right) + \tfrac{C_{i-1}(L)+A_{i-1}(L)}{L(1-q)}\left(\tfrac{C_{i-1}(L)}{A_{i-1}(L)}\right)\right) + C_i(L) \\
(L+1)qc_i(L+1) = {} & iqa_i(L)\left(-\tfrac{qc_i(L)}{(1-2q)a_i(L)}\right) + (i+1)qa_{i+1}(L)\left(\tfrac{qc_{i+1}(L)}{(i-2q)a_{i+1}}+1\right) + \\
& qa_1(L)\left(\tfrac{a_i(L)}{\sum a_j(L)}\right)\left(\tfrac{qc_1(L)}{(1-2q)a_1(L)}+1\right) + (qc_i(L)+i(1-2q)a_i(L))\left(-\tfrac{qc_i(L)}{(1-2q)a_i(L)}\right) + \\
& (qc_{i-1}(L)+(i-1)(1-2q)a_{i-1}(L))\left(\tfrac{qc_{i-1}(L)}{(1-2q)a_{i-1}(L)} + Lqc_i(L)\right. \\
(L+1)q\Delta c_i = {} & -c_i\left(\tfrac{iq^2}{1-2q} + \tfrac{q^2c_i}{(1-2q)a_i} + iq + q\right) + \tfrac{(i+1)q^2c_{i+1}}{1-2q} + c_{i-1}\left(\tfrac{q^2c_{i-1}}{(1-2q)a_{i-1}} + (i-1)q\right) + \\
& qa_i\left(\tfrac{qc_1+(1-2q)a_1}{(1-2q)\sum a_j}\right) + (i+1)qa_{i+1}
\end{aligned}$$

Let $v = \sum a_i$. After taking into account the special cases at $i=1$, $i=p$ and $i=2p-1$, and performing algebraic manipulation, we get:

Theorem 3 *For a merge-at-half B-tree that is parameterized by q, the steady state values of a_i and c_i, $1 \le i \le 2p-1$ is the solution to the following set of non-linear equations:*

$$\begin{aligned}
0 &= -(i(1-q)+(1-2q))a_i - qc_i + (i+1)qa_{i+1} && i=1 \\
0 &= -(i(1-q)+(1-2q))a_i - qc_i + (i+1)qa_{i+1} + qc_{i-1} + (i-1)(1-2q)a_{i-1} && \\
&\quad +2qc_{2p-1} + 2(2p-1)(1-2q)a_{2p-1} && i=p \\
0 &= -(i(1-q)+(1-2q))a_i - qc_i + qc_{i-1} + (i-1)(1-2q)a_{i-1} && i=2p-1 \\
0 &= -(i(1-q)+(1-2q))a_i - qc_i + (i+1)qa_{i+1} + qc_{i-1} + (i-1)(1-2q)a_{i-1} && \text{otherwise}
\end{aligned}$$

$$\begin{aligned}
0 &= -c_i(iq + qc_i/a_i + (1-2q)(i+1)) + (i+1)qc_{i+1} && \\
&\quad +\tfrac{a_i}{v}(qc_1 + (1-2q)a_1) + (1-2q)(i+1)a_{i+1} && i=1 \\
0 &= -c_i(iq + qc_i/a_i + (1-2q)(i+1)) + (i+1)qc_{i+1} + c_{i-1}(qc_{i-1}/a_{i-1} + (1-2q)(i-1)) + &&
\end{aligned}$$

$$\begin{array}{lll} & \frac{a_1}{v}(qc_1 + (1-2q)a_1) + (1-2q)(i+1)a_{i+1} + c_{2p-1}(\frac{qc_{2p-1}}{a_{2p-1}} + (1-2q)(2p-1)) & i = p \\ 0 = & -c_i(iq + qc_i/a_i + (1-2q)(i+1)) + c_{i-1}(qc_{i-1}/a_{i-1} + (1-2q)(2p-2)) & \\ & +\frac{a_1}{v}(qc_1 + (1-2q)a_1) & i = 2p-1 \\ 0 = & -c_i(iq + qc_i/a_i + (1-2q)(i+1)) + (i+1)qc_{i+1} + c_{i-1}(\frac{qc_{i-1}}{a_{i-1}} + (1-2q)(i-1)) + & \\ & \frac{a_1}{v}(qc_1 + (1-2q)a_1) + (1-2q)(i+1)a_{i+1} & \text{otherwise} \end{array}$$

$$0 = \quad 1 - \sum_{j=1}^{2p-1} c_i$$

The equations for the equilibrium point of the Δc_i are linearly dependent, as again they sum to zero. In order to solve the system, remove one of the Δc_i equations, which gives $4p-2$ equations in $4p-2$ variables.

Note that if $q = .5$, then we have the pure modify model, and if $q = 0$, then we have the pure insert model. Thus this model generalizes both the pure-modify ghost model and Yao's pure-insert model.

If we add together the equations for the equilibrium points of the Δa_i, we get the relation:

$$0 = -(1-2q)\sum_{i=1}^{2p-1} a_i - a_1 q + (1-2q)(2p-1)a_{2p-1} + qc_{2p-1}$$

The formula can be interpreted as follows: On a delete operation, the probability of merging a node is a_1. Since q is the probability of an operation being a delete, $qa_1 = R_d =$ the rate at which nodes are merged, in nodes per operation. Similarly, $(1-2q)(2p-1)a_{2p-1} + qc_{2p-1} = R_s =$ the rate at which nodes split. If we let $R_s - R_d = R_g =$ the rate at which nodes are added to the B-tree, and we recall the relationship between U and $\sum a_i$, we have the relationship:

$$U = \frac{1-2q}{(2p-1)R_g}$$

As we shall see in the next section, U remains stable over a wide range of q if p is large. Since R_d is negligible compared to R_s if $q < .5$, the relationship can be used as a rule of thumb to calculate the probability of splitting.

The equations were solved for $q = .45$ and $q = .47$, and p varying between 5 and 20. Table 2 shows a comparison between analytical results and simulation results.

Unfortunately, these equations are difficult to solve, as non-linear equation solvers require a good estimate of the starting point. We calculated the solutions for only a few cases. This points out the need for an approximation that can be easily solved.

2.3.2 Linear Model

The difficulty in solving the ghost model of a B-tree is that the ghosts satisfy a non-linear recurrence. Ghosts are necessary when the number of ghosts in a node outnumber the number of items. If q is small, however, then the number of items in a node is greater than the number of ghosts in a node. In addition, the distribution of the ghosts among the nodes of different order has a similar distribution to the number of items. Therefore, we can try making the approximation that the probability that a node of order i receives an insert is ia_i. Using this approximation gives us linear equations, so we will call this model the *linear model*. Mizoguchi described a similar model for B-trees with even maximum node size [Mizoguchi 79]. In this section, we will describe the linear model using the methods that we have developed, examine its range of accuracy, then examine the results on B-tree utilization.

The linear model is described by the following recurrence:

$$A_i(L+1) = \quad q\left[(1 - \frac{i}{(1-2q)L}A_i(L) + \frac{i+1}{(1-2q)L}A_{i+1}(L)\right]$$

$$+(1-q)\left[(1-\tfrac{i}{(1-2q)L})A_i(L)+\tfrac{i-1}{(1-2q)L}A_{i-1}(L)\right]$$

If we then solve for ΔP_i, where $P_i = ia_i$, we get:

Theorem 4 *Under the approximation that a node of order i receives an insert is ia_i, the P_i are the solution to the following set of linear equations:*

$$\begin{array}{rcl}
0 = & -(1+(1-2q))P_1 + qP_2 & i=1 \\
0 = & -(p+(1-2q))P_p + (1-q)pP_{p-1} + qpP_{i+1} + 2p(1-q)P_{2p-1} & i=p \\
0 = & -(2p-1+(1-2q))P_{2p-1} + (1-q)(2p-1)P_{2p-2} & i=2p-1 \\
0 = & -(i+(1-2q))P_i + (1-q)iP_{i-1} + qiP_{i+1} & \text{otherwise} \\
0 = & 1-\sum_{i=1}^{2p-1} P_i &
\end{array}$$

The existence, uniqueness and stability of the equilibrium point holds by the previous argument.

Tables 4, 5 and 6 show a comparison between the simulation results and the analytic results. Examining the tables, we see that the linear approximation is very accurate for $q < .4$. At $q = .45$, the ghost model gives results that more closely match those of the simulation. The case of $q = .5$ gives the merge-at-empty pure-modify models of [Mizoguchi 79] and [Quitzow 80]. The a_i calculated from the model for $q = .5$ is plotted along with the simulation and ghost model results in figure 1. As can be seen, the linear model does not calculate the distribution of nodes for $q = .5$.

If $q < .5$ however, the linear model becomes more accurate as p increases. In other words, when inserts significantly outnumber deletes and the maximum node size increases, the linear model becomes more accurate. This happens for two reasons: as p increases the distribution of the c_i becomes more similar to the distribution of the ia_i; also there are fewer very small nodes, the nodes for which ia_i is the poorest approximation to c_i.

Figure 3 shows a comparison between the ghost model, the linear model and the simulation. Notice that the linear model always gives low estimates of the utilization, even for the pure-modify case. The low estimates of the utilization occur because the linear model underestimates the probability that a small node receives an insert and thus becomes larger (a small node must have received many deletes). This means that we can use the linear model to provide a lower bound on the utilization of a random B-tree.

Figure 4 shows how the utilization varies with p for different q. Notice that as p becomes large, the utilization curve becomes flatter with a sharper knee near $q = .5$. If we examine the entries in Table 5 for $p = 100$, we see that the linear model predicts a utilization of 65.48% even when $q = .47$, which is a lower bound by the previous paragraph.

While the linear model gives reasonable estimates of the space utilization, it gives very poor estimates of the probability of splitting or merging when q becomes close to .5. The linear model predicts a quadratic decrease in P_s and P_m as p becomes larger for pure-modify operations, but the ghost model and the simulation show an exponential decrease. Since the linear model underestimates the probability that a small node receives an insert, the linear model overestimates the probability of merging on a delete (the simulation and the ghost model predict that P_m is almost zero if $q < .5$). Because the predicted utilization is close to the actual utilization, the rule-of-thumb requires that the overestimate of merges must be balanced by an overestimate of splits. However, if q is small enough ($q < .4$), then the linear model gives good estimates of P_m and P_s.

3 Merge-at-Half B-trees

We next analyze merge-at-half B-trees. The analysis of merge-at-half B-trees is much more difficult than the analysis of merge-at-empty B-trees because a half-empty node may merge with any other node.

Therefore, we now present only an approximate analysis for the purpose of comparing to merge-at-empty B-trees. The ghost model of merge-at-half B-trees is presented in the appendix. The following analysis is similar to the analysis of [Quitzow 80], who model the B-tree nodes using differential equations. However, [Quitzow 80] examines only the cases of pure-insert and pure-modify. For a comparison, we need to know the utilization for the entire range of q, because merges might (and in fact do) cause the utilization to increase when there are deletes in the instruction mix. Also, [Quitzow 80] does not calculate the restructuring probabilities.

3.1 Analysis

The first step in the analysis is to specify how the tree is modified on an insert or a delete. The action on an insert is the same that for a merge-at-empty B-tree.

$$\begin{aligned}(X_p, \ldots, X_i, X_{i+1}, \ldots, X_{2p-1}) &\rightarrow \quad (X_p, \ldots, X_i - 1, X_{i+1} + 1, \ldots, X_{2p-1})\\ (X_p, \ldots, X_{2p-1}) &\rightarrow \quad (X_p + 2, \ldots, X_{2p-1} - 1)\end{aligned}$$

The action on a delete is also the same as that for a merge-at-empty B-tree if no node gets merged (no delete from an order p node).

$$(X_p, \ldots, X_i, X_{i+1}, \ldots, X_{2p-1}) \rightarrow (X_p, \ldots, X_i + 1, X_{i+1} - 1, \ldots, X_{2p-1})$$

If a delete is directed to an order p node, then the node gets merged with its neighbor. Thus the action of a delete that causes a merge depends on the sibling of the merged node. Suppose that the sibling is an order j node. Then

$$\begin{array}{lll}(X_p, \ldots, X_{2p-1}) \rightarrow & (X_p - 2, \ldots, X_{2p-1} + 1) & j = p\\ \quad (X_p, \ldots, X_{(p+j-1)/2}, \ldots, X_j, \ldots, X_{2p-1}) \rightarrow & & \\ & (X_p - 1, \ldots, X_{(p+j-1)/2} + 2, \ldots, X_j - 1, \ldots, X_{2p-1}) & p + j - 1 \text{ is even}\\ \quad (X_p, \ldots, X_{(p+j)/2-1}, X_{(p+j)/2}, \ldots, X_j, \ldots, X_{2p-1}) \rightarrow & & \\ & (X_p - 1, \ldots, X_{(p+j)/2-1} + 1, X_{(p+j)/2} + 1, \ldots, X_j - 1, \ldots, X_{2p-1}) & p + j - 1 \text{ is odd}\end{array}$$

Note that not every node order has the chance of gaining on a merge operation. In particular, if p is even, then $(3p-2)/2$ is the largest order of nodes that will gain; if p is odd, then $(3p-1)/2$ is the largest order of nodes that will gain.

Since the structure of the evolution of the B-tree is different if p is odd or even, let us examine first the case when p is even. An order i node will be merged with its neighbor if its neighbor is of order p and receives a delete (assume a node merges with its right neighbor, except if the node is the rightmost node, in which case it merges with its left neighbor). The probability that a node of order i is the neighbor of the merging node is $A_i(L)/V(L)$. The result of the merge operation is two nodes of order $(i+p-1)/2$, or a node of order $(i+p)/2$ and a node of order $(i+p-2)/2$. Therefore, one node of order j will be created if a node of order $2j-p$ or a node or order $2i-p+2$ is merged, and two nodes of order i will be created if a node of order $2j-p+1$ is merged. If $j = (3p-2)/2$, then $2j-p+2 = 2p$, so nodes of order $(3p-2)/2$ can't be created from merges involving nodes of order $2j-p+2$. If the neighboring node involved in the merge is of order p, then a node of order $2p-1$ is created. So, if $j = p$, then nodes of order j aren't created from merges with nodes of order $2j-p = p$.

In order to simplify the analysis, approximate the probability that an insert is directed to an order i node by

$$\frac{iA_i(L)}{L(1-2q)}$$

Using the approximation to the probability of inserting at an order i node, we have:

Theorem 5 *If p is even and we use the linear approximation, then in a merge-at-half B-tree the a_i, $1 \le i \le p$ satisfy the following set of nonlinear equations:*

$$\begin{array}{rll}
0 = & -(p+(1-2q))a_p + q(p+1)a_{p+1} + 2(1-q)(2p-1)a_{2p-1} & \\
& +\frac{qpa_p}{v}(-a_p + 2a_{p+1} + a_{p+2}) & i = p \\
0 = & -(i+(1-2q))a_i + q(i+1)a_{i+1} + (1-q)(i-1)a_{i-1} & \\
& +\frac{qpa_p}{v}(-a_i + a_{2i-p} + 2a_{2i-p+1} + a_{2i-p+2}) & p < i < \dfrac{3p-2}{2} \\
0 = & -(i+(1-2q))a_i + q(i+1)a_{i+1} + (1-q)(i-1)a_{i-1} & \\
& +\frac{qpa_p}{v}(-a_i + a_{2i-p} + 2a_{2i-p+1}) & i = \dfrac{3p-2}{2} \\
0 = & -(i+(1-2q))a_i + q(i+1)a_{i+1} + (1-q)(i-1)a_{i-1} & \\
& -\frac{qpa_pa_i}{v} & 3p/2 \le i < 2p-1 \\
0 = & -(i+(1-2q))a_i + (1-q)(i-1)a_{i-1} & \\
& +\frac{qpa_p}{v}(-a_i + a_p) & i = 2p-1 \\
0 = & 1 - \sum_{j=p}^{2p-1} ia_i &
\end{array}$$

When p is odd, the equations that describe the a_i are the same as when p is even, except that that $a_{(3p-1)/2}$ is the highest order node that can gain on a merge, and only from an order $2i-p$ node:

Theorem 6 *If p is odd and we use the linear approximation, then in a merge-at-half B-tree the a_i, $1 \le i \le p$ satisfy the following set of nonlinear equations:*

$$\begin{array}{rll}
0 = & -(p+(1-2q))a_p + q(p+1)a_{p+1} + 2(1-q)(2p-1)a_{2p-1} & \\
& +\frac{qpa_p}{v}(-a_p + 2a_{p+1} + a_{p+2}) & i = p \\
0 = & -(i+(1-2q))a_i + q(i+1)a_{i+1} + (1-q)(i-1)a_{i-1} & \\
& +\frac{qpa_p}{v}(-a_i + a_{2i-p} + 2a_{2i-p+1} + a_{2i-p+2}) & p < i < \dfrac{3p-1}{2} \\
0 = & -(i+(1-2q))a_i + q(i+1)a_{i+1} + (1-q)(i-1)a_{i-1} & \\
& +\frac{qpa_p}{v}(-a_i + a_{2i-p}) & i = \dfrac{3p-1}{2} \\
0 = & -(i+(1-2q))a_i + q(i+1)a_{i+1} + (1-q)(i-1)a_{i-1} & \\
& -\frac{qpa_pa_i}{v} & \dfrac{3p-1}{2} < i < 2p-1 \\
0 = & -(i+(1-2q))a_i + (1-q)(i-1)a_{i-1} & \\
& +\frac{qpa_p}{v}(-a_i + a_p) & i = 2p-1 \\
0 = & 1 - \sum_{j=p}^{2p-1} ia_i &
\end{array}$$

The a_i are also subject to the condition that $\sum ia_i = 1$. We solved the above set of non-linear equations with the numerical analysis package NAG [NAG].

3.2 Comparison

We modified the merge-at-empty simulator to make a merge-at-half B-tree and performed experiments to compare against the results of our analysis. In tables 8, 9 and 10, we compare the results from the analysis and the simulations.

The utilization predicted by the analysis and by the simulation agree well, although the difference increases as q approaches .5. Surprisingly, the space utilization increases as deletes become more common in the instruction mix. The utilization increases because a merging node will increase in size. However, the increase in utilization is small, as the utilization never goes above 71%.

The space utilization stays level for most values of the parameter q, but decreases sharply as q approaches .5. Furthermore, for $q = .5$, the space utilization decreases as p increases. Both the analysis and the simulation indicate that the utilization will approach about 60%.

The analysis and the simulation do not agree as well on the probability of splitting or merging. The difference is small when $q = .1$, but becomes larger as q increases. The increasing error is due to ignoring the effect of ghosts, whose effect becomes more significant as q increases. The analysis consistently underestimates the probability of splitting or merging.

Next, we use the results on merge-at-empty B-trees from the previous section to compare merge-at-half B-trees against merge-at-empty B-trees. For the comparison, we used a B-tree with $p = 40$. In Figure 7, we compared the utilization of a merge-at-half and a merge-at-empty B-tree. The figure shows that the utilization of both B-trees remains close for most values of the parameter q. Up to $q = .45$ the utilization of the merge-at-half B-tree is less than 10% greater than the utilization of the merge-at-empty tree, but when $q = .5$, the difference becomes 60%

In Figure 8, we compare the probability of restructuring on an operation ($= q\,\Pr[\text{merge on delete}]+(1-q)\Pr[\text{split on insert}]$). The probability of restructuring a merge-at-half B-tree is 20% greater when $q = .1$ and 300% greater when $q = .45$. When $q = .5$, the rates cannot be compared because the probability of restructuring a merge-at-empty B-tree becomes infinitesimal, but the probability of restructuring a merge-at-half tree becomes large. Table 11 also compares restructuring rates for several values of the parameter p.

4 Upper Levels

The upper-level structure of random B-trees is important for analyzing the performance of concurrent B-tree algorithms. We will analyze two kinds of restructuring algorithms; *bottom up* and *top down*. The bottom-up algorithm splits a non-leaf node only if it is full and one of its children splits. The top-down algorithm splits any full non-leaf node that an insert encounters on its path to a leaf. At the leaf level, we will assume that the B-tree uses merge-at-empty. Since merges are rare with merge-at-empty, we will ignore them.

A leaf will be defined to be on level 1. The parent of the leaf is on level 2, and the root is on level l.

4.1 Bottom-Up Algorithm

In this analysis, we assume that nodes are merged only when they become empty, and that no preparatory operation are performed. We have established that ghosts are necessary for modeling B-trees in the presence of deletes. However, we have also shown that if p is large and $q < .5$, then the merge rate is vanishingly small compared to the split rate. We will first deal with the case $q < .5$ now, and handle the case $q = .5$ later. Therefore, there will be very few second level ghosts as compared to the number of second level entries, so we do not need to model second level ghosts.

Let $A_{2,i}(N)$ be the expected number of second level, order i nodes. Let $E_1(N, q)$ be the expected number of items in a leaf-level node. We have shown that $E_1(N, q)$ is independent of N, so we will drop the dependence on N in the notation from now on. After N modify operations, each of which is a delete with probability q and is an insert with probability $1 - q$, the expected number of items in the B-tree is $N(1 - 2q)$. Using the same probability distribution as for the leaf level, the probability that a insert is directed to a node is proportional to the number of items and ghosts that the node covers. Since first level merge operations are rare, deletes on the second level are rare, so ghosts on the second level have negligible effect. If we assume that the structure of the first level is independent of the structure of the second level, then the number of items and ghosts covered by a node is proportional to the size of the node. Therefore, the probability that an insert is directed to an order i, level 2 node is:

$$\Pr[I_{2,i}] = \frac{iA_{2,i}(N)E_1(q)}{N(1-2q)}$$

A level 2 order i nodes becomes an order $i+1$ node (or becomes 2 order p nodes) only if a level 1 node splits. The probability that a node splits is $(1-2q)/((E_1(q)(1-q))$. Therefore, the probability that a level 2 order i node gets inserted into on an operation is $iA_{2,i}(N)/(N(1-q))$. Since an operation is an insert with probability $1-q$, and propagated deletes are rare, the equations that describe the second level nodes in a merge-at-empty B-tree are the equations that describe the pure-insert leaves. We can reach the same conclusion for all non-leaf levels in the B-tree by inductively applying the above arguments.

We simulated a B-tree and ran experiments to compare the resulting $a_{2,i}$ against the prediction. The B-tree was given a parameterized mixture of insert and delete operations until it grew to contain 120,000 items. 12 snapshots of the second level of the B-tree were collected. The snapshots were averaged together to obtain the results in Table 12.

Pure Modify ($q = .5$) For the case of $q = .5$, or pure modify, we have shown that almost no restructuring operations are performed. The size of the B-tree does not increase if $q = .5$, so the tree must have grown from an instruction mix where $q < .5$. Since the previous section shows that the structure of the second level of a B-tree is independent of q, except for scaling, the above results hold for the case of $q = .5$ in the short term. Eventually, all levels of the pure-merge B-tree will have a leaf-level pure-modify structure, but since the probability of merging a node on a delete is less than .00002 if $p = 20$, and a large number of modifies is needed to before the equilibrium state is reached, the upper levels of even pure-modify B-trees will have a pure-insert distribution of nodes sizes in practice.

4.2 The Top-Down Tree

4.2.1 Second Level Analysis

In a top-down B-tree, preparatory operations are performed on the path from the root to the leaf. An insert operation that comes across a a full node splits the node and a delete operation that comes across a near-empty node merges it. Since propagated merge operations are rare, we will concentrate on the effect of the preparatory split operations.

Since preparatory operations will split a full node, the B-tree model must be slightly modified: a full node will split into a node of order p and a node of order $p-1$.

Let $I_{2,i}$ be the event that an insert operation passes through a level 2, order i node on its path to a leaf. Let $S_{2,i}$ be the event that an insert operation propagates a split to an order i, level 2 node. Then:

$$\Pr[I_{2,i}] = \frac{iA_{2,i}(N)E_1(q)}{N(1-2q)}$$

For $i = p-1, \ldots, 2p-2$, $\Pr[S_{2,i}] = p_{1,s} \Pr[I_{2,i}]$, so

$$\Pr[S_{2,i}] = \frac{1-2q}{1-q} i a_{2,i}(N) \quad i = p-1, \ldots, 2p-2$$

Since a full node is split every time an insert operation reaches it,

$$\Pr[S_{2,2p-1}] = (2p-1)a_{2,2p-1}(N)E_1(q)$$

The recurrence equations that describe the the $a_{2,i}$ are:

$$A_{2,i}(N+1) = \quad (1-q)(A_{2,i}(N) - (1)\tfrac{iA_{2,i}(N)}{(1-q)N} + (1)\tfrac{(i-1)A_{2,i-1}(N)}{(1-q)N}) + q(A_{2,i}(N) - (1)(0) + (1)(0))$$

except for $i = p-1$, $i = p$, and $i = 2p-1$. For $i = 2p-1$, the recurrence equation is:

$$\begin{aligned} A_{2,2p-1}(N+1) = \;& (1-q)\left(A_{2,2p-1}(N) - (2p-1)E_1(q)a_{2,2p-1}(N) + \tfrac{1-2q}{1-q}(2p-2)a_{2,2p-1}(N)\right) \\ & +qA_{2,2p-1}(N) \end{aligned}$$

After deriving the equations for $a_{2,p-1}$ and $a_{2,p}$ and solving for the equilibrium point, we get the system:

$$\begin{aligned} 0 &= -pa_{2,p} + \left(\tfrac{1-q}{1-2q}(2p-1)E_1(q)\right)a_{2,2p-1} \\ 0 &= -(p+1)a_{2,p} + (p-1)a_{2,p-1} + \left(\tfrac{1-q}{1-2q}(2p-1)E_1(q)\right)a_{2,2p-1} \\ 0 &= -(i+1)a_{2,i} + (i-1)a_{2,i-1} \qquad i = p+1,\ldots,2p-1 \\ 0 &= -\left[\tfrac{1-q}{1-2q}(2p-1)E_1(q)+1\right]a_{2,2p-1} + (2p-2)a_{2,2p-2} \end{aligned}$$

In order to make the system of equations more tractable, make the substitution $P_{2,i} = ia_{2,i}$:

$$\begin{aligned} 0 &= -pP_{2,p-1} + \left(\tfrac{1-q}{1-2q}(p-1)E_1(q)\right)P_{2,2p-1} \\ 0 &= -(p+1)P_{2,p} + pP_{2,p-1} + \left(\tfrac{1-q}{1-2q}pE_1(q)\right)P_{2,2p-1} \\ 0 &= -(i+1)P_{2,i} + iP_{2,i-1} \qquad i = p+1,\ldots,2p-1 \\ 0 &= -\left[\tfrac{1-q}{1-2q}(2p-1)E_1(q)+1\right]P_{2,2p-1} + (2p-1)P_{2,2p-2} \end{aligned}$$

Let $\alpha = ((1-q)/(1-2q))(2p-1)E_1(q)+1$. Then, derive the formulae for the P_i as in the pure-insert case.

$$\begin{aligned} P_{2,i} &= \tfrac{\alpha}{i+1}P_{2,2p-1} \qquad i = p,\ldots,2p-2 \\ P_{2,p-1} &= \left(\tfrac{(p-1)(1-q)}{p(1-2q)}E_1(q)\right)P_{2,2p-1} \end{aligned}$$

subject to

$$\sum_{i=p-1}^{2p-1} P_{2,i} = 1/E_1(q)$$

Summing the $P_{2,i}$, we get:

$$\begin{aligned} P_{2,2p-1} &= \left(E_1(q)\left[\left(\tfrac{1-q}{1-2q}(2p-1)E_1(q)+1\right)(H(2p-1)-H(p)) + 1 + \tfrac{(p-1)(1-q)}{p(1-2q)}E_1(q)\right]\right)^{-1} \\ P_{2,p-1} &= \left(\tfrac{(p-1)(1-q)}{p(1-2q)}E_1(q)\right)P_{2,2p-1} \\ & i = p,\ldots,2p-2 \\ P_{2,i} &= \left(\tfrac{(1-q)(2p-1)}{1-2q}E_1(q)+1\right)\left(\tfrac{P_{2,2p-1}}{i+1}\right) \end{aligned}$$

We simulated a B-tree with preparatory operations and ran experiments to compare the resulting $a_{2,i}$ against the prediction. The B-tree was given a parameterized mixture of insert and delete operations until it grew to contain 120,000 items. 12 snapshots of the second level of the B-tree were collected. The snapshots were averaged together to obtain the results in Table 13.

The values of p and q have an effect primarily on nodes of order $2p-1$. From the formula for $P_{2,2p-1}$, we can see that as q approaches .5, $1/(2q-1)$ becomes very large, so that $P_{2,2p-1}$ becomes very small. Also, $P_{2,2p-1}$ depends on $1/E_1^2(q)$, so that $P_{2,2p-1}$ the ratio of $P_{2,2p-1}$ and $P_{2,i}$ decreases as $1/p$. Therefore, if p is large, full upper level nodes are rare in a top-down B-tree, a desirable characteristic for a concurrent B-tree.

4.2.2 Level 3 and Beyond

The analysis of the third and higher levels of a top-down B-tree proceeds as the second level analysis does. inductively. We find that:

$$\begin{aligned}\Pr[S_{2,i}] &= \frac{1-2q}{1-q} i a_{3,i}(N) \qquad i = p, \ldots, 2p-1 \\ &= (2p-1)a_{3,(2p-1)}(N)E_1(q)E_2(q)\end{aligned}$$

Thus, the equations that describe the $P_{3,i}$ are the same as those that describe the $P_{2,i}$, except with $E_1(q)E_2(q)$ substituted for every occurrence of $E_1(q)$.

The space utilization of the higher levels of a top-down tree can be calculated using Lemma 2. Table 14 shows the utilization of several levels of a top-down tree for various values of q, and assuming $p = 30$.

4.3 Root Analysis

Two important questions about the root of the B-tree are what level is the root on and how many children does it have. We will provide formulae for calculating these, but the answers will depend on the number of items in the B-tree.

4.3.1 Bottom-Up Algorithm

Let us begin by asking the question: suppose the root is on level h and has i children. What is the expected number of items in the tree?

Let $N_R(h,i)$ be the number of items covered by an order i, level h root, and let $E_R(h,i) = E[N_R(h,i)]$. From section 4.1.2, a node on level j will hold an expected

$$\begin{aligned}E'_j &= E_j E_{j-1} \cdots E_1 \\ &= E^{j-1} E_1\end{aligned}$$

Where

$$E = 2p[H(2p) - H(p)]$$

If we assume that the children of the root have a steady-state distribution of node sizes, then a B-tree with a level h, order i root will hold an expected

$$E_R(h,i) = iE^{h-1}E_1$$

If, however, $i = 2$ and you know that the root has recently split then the expected number of items in the B-tree is $E_R(h-1, 2*p)$, because the two children of the root should both be of order p.

Next, we want to turn these predictions around so that we can answer the question: if a B-tree contains N items, what is the level and the order of the root?

In order to answer this question, it is instructive to know the variance of $N_R(h,i)$, $V_R(h,i)$.

Our starting point is the following formula. Let N be a non-negative integer valued random variable and let X_i be independent random variables. Then ([Ross 84]):

$$V\left[\sum_{i=1}^{N} X_i\right] = E[N]V[X] + (E[X])^2 V[N]$$

Let V'_i be the variance of number of items covered by a level i node. Suppose you knew E'_{i-1} and V'_{i-1}. The number of items covered by a level h node is the sum of the number of items covered by its children.

The number of children of a level h node is a random variable $A(h)$, where $\Pr[A(h) = i] = a_{h,i}/(\sum a_{h,j})$. From section 5.2, the expected value and variance of $A(h)$, E and V, are:

$$\begin{aligned} E &= \frac{1}{\sum a_{h,j}} \\ &= 2p[H(2p) - H(p)] \\ V &= \sum_{i=p}^{2p-1} \frac{i^2 a_{h,i}}{\sum a_j} \\ &= 2p(p - [H(2p) - H(p)]) \end{aligned}$$

Therefore, we have the following recursive equation for E'_h and V_h:

$$\begin{aligned} E'_h &= EE'_{h-1} \\ V'_h &= EV_{h-1} + (E'_{h-1})^2 V \end{aligned}$$

Table 15 lists E'_h and V'_h for $p = 10$ and assuming pure inserts. The variance of the distribution is large compared to the mean.

With this information, we can try to approximate the number of items in the B-tree if the root is at a certain height and on a certain level. By using the mean and variance (and higher moments, if desired), one can compute bounds on the probability that the root is on a certain level and of a certain order. For a rougher estimate, a simpler mechanism can be used: For every (h, i), calculate $E_R(h, i)$. Assign to each (h, i) a partition of the positive integers by:

$2 < i \leq 2p - 1$: (h, i) is assigned the interval $[(E_R(h, i-1) + E_R(h, i))/2, (E_R(h, i) + E_R(h, i+1))/2]$

$i = 2$: (h, i) is assigned the interval $[(E_R(h-1, 2p-1) + E_R(h-1, 2p))/2, (E_R(h, 2) + E_R(h, 3))/2]$

In order to estimate the root height and order, find the interval that N falls in, and use the (h, i) assigned to the interval as the answer. Table 16 compares the (h, i) estimated by this method against root height and order from a simulated B-tree for various values of $E_R(h, i)$.

The above analysis assumes that the distribution of node sizes of the children of the root can be closely approximated by the steady-state distribution. However, the children of the root are new enough that the transients in the node size distribution have not died out. The transient distribution of the node sized of the children of the root is approximated in [Driscoll 87]. Although [Driscoll 87] uses an approximation to the transient distribution, it can more accurately track the size of the B-tree when a child of the root splits. The analysis in [Driscoll 87] assumes a pure-insert B-tree. Since the upper levels of a B-tree with inserts and deletes acts like a pure-insert B-tree that grows at the net-insert rate, Driscoll's analysis applies to a parameterized B-tree also.

4.3.2 Top-Down Root Analysis

The root of a top-down tree can be analyzed in the same way as above, except values of $E_h(q)$ and $V_h(q)$ must be used, and $E_R(h, 2) = E_R(h-1, 2p-1)$ if the root has recently split. Notice that E and V are now dependent on both h and q. Table 17 compares the estimated (h, i) against a simulation for various values of $E_R(h, i)$.

5 Conclusion

This paper has presented methods to calculate the equilibrium utilization and distribution of nodes in a B-tree when the probability of an operation being a delete ranges between 0 and .5. The random B-tree is modeled by a set of simultaneous recurrence equations. By transforming the recurrence equation into a difference equation and searching for the equilibrium point, the problem is transformed into finding the solution of a set of equations.

In order to model the situation when deletes are allowed into the operation mix, the notion of a ghost (an item that was deleted from the B-tree) was used. The distribution of ghosts in the B-tree affects the input probability distribution; thus it must be calculated along with the distribution of the nodes in the B-tree. The ghost model achieved accurate calculations: less than 5% error when $q \leq .45$ and $p \geq 20$ in calculating both the utilization and the probability of splitting on an insert.

As q approaches .5, the distribution of nodes and ghosts in the B-tree becomes harder to calculate accurately. When $q = .47$, the error in calculating the utilization and the probability of splitting becomes 10%. When $q = .5$, the error in calculating the utilization is within 10%, but the analytical probability of splitting is inaccurate. However, the simulation agrees with the model's prediction that the probability of splitting or merging is very small and decreases exponentially with p.

The major problem with the ghost model is that it is hard to solve. An approximation to the ghost model is presented that reduces the problem of calculating the utilization and probability of splitting and merging in a random B-tree to solving a set of simultaneous linear equations. This model is very accurate for a wide range of the parameter q. However, when q becomes larger than .4, the linear model becomes inaccurate and the ghost model must be used.

5.1 Pragmatic Conclusions

The calculations of the utilization of random merge-at-empty B-trees parameterized by p and q shows that U remains between 60% and 70% for most values of q ($\leq .45$) if p is large (≥ 15), but drops to 39% when $q = .5$, or if all operations are modifies. Trees in which deletes outnumber inserts are rarely interesting in practice and are difficult to model in the limit. The knee of the utilization curve gets closer to $q = .5$ and becomes sharper as p becomes larger.

The tendency of the utilization to remain near 69% can be explained by the following arguments: If there are even just a few more inserts than deletes, the B-tree will grow at the net insert rate (the rate of inserts minus the rate of deletes). Furthermore, nodes with more items are more likely to get hit by a delete than smaller nodes, so that 1) it is hard for small nodes to become smaller and 2) larger nodes tend to bunch up near 69%. These trends can be seen in Figure 6. The curve has the same shape as a curve for a pure-insert B-tree. The number of nodes with less than p items decreases exponentially as p decreases, and thus has little effect on the expected node size. Since the largest nodes are the most likely to get hit with a delete, the largest nodes are pushed downwards. Fewer nodes get split so fewer half-full nodes appear, and half-full nodes tend to get pushed upwards. These tendencies become stronger as p increases, causing the utilization to stay near the pure insert utilization even as q approaches .5.

The simulation and the ghost model show that the probability of merging a node on a delete in a merge-at-empty B-tree is almost zero. From the parameterized ghost model, we can relate the utilization of the B-tree and the probability of splitting on an insert by the formula:

$$P_s = \frac{1 - 2q}{(1 - q)(2p - 1)U}.$$

This follows from the relation between the rate of growth and the utilization derived in section 5.1 by assuming that the number of merges is negligible and adjusting the formula to 'per insert' instead of 'per operation'.

As it stands, the rule of thumb is not useful unless we have either the utilization or the probability of splitting available. In Figures 3 and 4, we see that the utilization tends to remain near the pure-insert level until q is approximately .47, so we can estimate the utilization to be 68% regardless of p and q. Figure 5 compares the probability of splitting derived from the rule of thumb against the simulation for varying q with $p = 30$ and $p = 40$. As can be seen, the rule of thumb gives very good estimates of the probability of splitting up to at least $q = .45$.

The tendency of merge-at-empty B-tree utilization to remain near the pure-insert level suggests that merging B-tree nodes when they are empty is not a wasteful strategy in terms of storage and is significantly better in terms of restructuring.

The upper levels of a merge-at-empty bottom-up restructuring B-tree have the same distribution of node sizes that the leaf-level of a pure-insert has, as intuition would suggest. The simple structure of the upper levels of a merge-at-empty B-tree allows a simple but accurate calculation the root level and order.

In this paper, we also solved an approximate model of a merge-at-half B-tree. The simplified model seriously underestimates the probability of restructuring. However, for the purpose of comparison to a merge-at-empty B-tree, the underestimation is not a problem.

A merge-at-half B-tree will always have a space utilization of at least 50%. When all operations are modify operations, or when the number of insert operations is the same as the number of delete operations, then the utilization will be about 60%. In contrast, a merge-at-empty B-tree has a 0% lower bound on its space utilization, and will have about 39% utilization on a pure-modify instruction mix. However, the space utilization of a merge-at-empty B-tree remains high if there are just a few more insert operations than delete operations. Thus, merge-at-half usually buys little in terms of space utilization.

In Figure 8, we showed that the restructuring rate of a merge-at-half B-tree is significantly larger than the restructuring rate of a merge-at-empty B-tree for all values of $q \geq .1$. For many concurrent B-tree algorithms used in practice ([Bayer 77]), restructuring causes a serialization bottleneck. Thus, one simple but important way to increase concurrency in B-tree operations is to reduce the probability of restructuring. Since merge-at-half buys little space utilization but is expensive in terms of restructuring, we recommend that B-trees (especially concurrently accessed ones) use merge-at-empty.

6 Acknowledgement

We would like to thank Jeanette Schmidt for helpful insights concerning ghosts.

7 Appendix

7.1 Merge-at-Half Ghost Model

In this section, we will show how to analyze a merge-at-half B-tree using ghosts.

In order to develop the recurrence equations that describe the expected number of nodes of various orders, we need to know what the expected change in the node size is on each step. Table 17 lists the possible changes in node size.

Using the information listed in Table 17, we can write down the recurrence equation for $A_i(L)$, $p < i < (3p-2)/2$:

$$\begin{aligned}
A_i(L+1) = \quad & A_i(L) - \frac{qiA_i(L)}{L(1-2q)} - (1-q)\frac{C_i(L)+iA_i(L)}{L(1-q)} - q\frac{pA_p(L)}{L(1-2q)}\frac{A_i(L)}{V(L)} \\
& +q\frac{(i+1)A_{i+1}(L)}{L(1-2q)} + (1-q)\frac{C_{i-1}(L)+(i-1)A_{i-1}(L)}{L(1-q)} + q\frac{pA_p(L)}{L(1-2q)}\frac{A_{2i-p}(L)}{V(L)} \\
& +2q\frac{pA_p(L)}{L(1-2q)}\frac{A_{2i-p+1}(L)}{V(L)} + q\frac{pA_p(L)}{L(1-2q)}\frac{A_{2i-p+2}(L)}{V(L)}
\end{aligned}$$

$$\begin{aligned}
(L+1)(1-2q)a_i(L+1) = \\
& L(1-2q)a_i(L) - qia_i(L) - qc_i(L) - (1-2q)ia_i(L) - qpa_p(L)\frac{a_i(L)}{v(L)} \\
& +q(i+1)a_{i+1}(L) + qc_{i-1}(L) + (i-1)(1-2q)a_{i-1}(L)
\end{aligned}$$

gain/loss	probability	range	cause
-1	$\frac{qiA_i(L)}{L(1-2q)}$	$i = p, \ldots, 2p-1$	delete
-1	$(1-q)\frac{C_i(L)+iA_i(L)}{L(1-q)}$	$i = p, \ldots, 2p-1$	insert
-1	$q\frac{pA_p(L)}{L(1-2q)}\frac{A_i(L)}{V(L)}$	$i = p, \ldots, 2p-1$	merge
1	$q\frac{(i+1)A_{i+1}(L)}{L(1-2q)}$	$i = p, \ldots, 2p-2$	delete
1	$(1-q)\frac{C_{i-1}(L)+(i-1)A_{i-1}(L)}{L(1-q)}$	$i = p+1, \ldots, 2p-1$	insert
2	$(1-q)\frac{C_{2p-1}(L)+(2p-1)A_{2p-1}(L)}{L(1-q)}$	$i = p$	insert
1	$q\frac{pA_p(L)}{L(1-2q)}\frac{A_{2i-p}(L)}{V(L)}$	$i = p+1, \ldots, \frac{3p-2}{2}$	merge
2	$q\frac{pA_p(L)}{L(1-2q)}\frac{A_{2i-p+1}(L)}{V(L)}$	$i = p, \ldots, \frac{3p-2}{2}$	merge
1	$q\frac{pA_p(L)}{L(1-2q)}\frac{A_{2i-p+2}(L)}{V(L)}$	$i = p, \ldots, \frac{3p-2}{2} - 1$	merge
1	$q\frac{pA_p^2(L)}{L(1-2q)V(L)}$	$i = 2p-1$	merge

Table 18: Changes in values of A_i on an operation

$$
\begin{aligned}
& +q\tfrac{pa_p(L)}{v(L)}(a_{2i-p}(L) + 2a_{2i-p+1}(L) + a_{2i-p+2}(L)) \\
\Delta(L+1)(1-2q)a_i(L) = \quad & -(i(1-q) + (1-2q))a_i(L) \\
& -qc_i(L) + q(i+1)a_{i+1}(L) + qc_{i-1}(L) \\
& +(i-1)(1-2q)a_{i-1}(L) + \tfrac{qpa_p(L)}{v(L)}(-a_i(L) + a_{2i-p}(L) + 2a_{2i-p+1}(L) + a_{2i-p+2}(L))
\end{aligned}
$$

Solving for the equilibrium point, we get, for $p < i < (3p-2)/2$:

$$
\begin{aligned}
0 = \quad & -(i(1-q) + (1-2q))a_i - qc_i + q(i+1)q_{i+1} + qc_{i-1} \\
& +(i-1)(1-2q)a_{i-1} + \tfrac{qpa_p}{v}(-a_i + a_{2i-p} + 2a_{2i-p+1} + a_{2i-p+2})
\end{aligned}
$$

The equilibrium point for the other cases can be found in a similar manner.

We next have to generate the equations for the ghosts. Table 18 lists the possible gains and losses The entries are similar to the entries in table A. The gain/loss caused by an insert/delete is the number of ghosts in the node, which is the total number of ghosts in order i nodes divided by the number of order i nodes $= C_i/A_i$. If a node of order $2i - p + 1$ merges with an order p node, then the number of ghosts contained in the order i nodes increases by the number of ghosts contained in the order p node plus the number of ghosts contained in the order $2i - p + 1$ node, plus the 1 ghost created. Assume that when two nodes of different sizes are created from a merge, the smaller of the two is the node that received entries. Assume also that the ghosts in a node are uniformly distributed between the items. Then a node of order i will receive $i/(2i-p)$ of the ghosts of an order $2i - p$ node, and a node of order i will receive the ghosts of the order p node, $(i-p+1)/(2i-p+2)$ of the ghosts of an order $2i-p+2$ node, and the ghosts that is created.

Using Table 18 to list the expected loss and gain on a single operation, we can generate the recurrence equation for $C_i(L)$, $p < i < (3p-2)/2$

$$
\begin{aligned}
C_i(L+1) = \quad & C_i(L) - \tfrac{C_i(L)}{A_i(L)}\left(q\tfrac{iA_i(L)}{L(1-2q)} + (1-q)\tfrac{C_i(L)+iA_i(L)}{L(1-q)} + q\tfrac{pA_p(L)}{L(1-2q)}\tfrac{A_i(L)}{V(L)}\right) \\
& +q\left(\tfrac{C_{i+1}(L)}{A_{i+1}(L)} + 1\right)\tfrac{(i+1)A_{i+1}(L)}{L(1-2q)} + (1-q)\tfrac{C_{i-1}(L)}{A_{i-1}(L)}\tfrac{C_{i-1}(L)+(i-1)A_{i-1}(L)}{L(1-q)} \\
& +q\tfrac{i}{2i-p}\tfrac{C_{2i-p}(L)}{A_{2i-p}(L)}\tfrac{pA_p(L)}{L(1-2q)}\tfrac{A_{2i-p}(L)}{V(L)} + q\left(\tfrac{C_p(L)}{A_p(L)} + \tfrac{C_{2i-p+1}(L)}{A_{2i-p+1}(L)} + 1\right)\tfrac{pA_p(L)}{L(1-2q)}\tfrac{A_{2i-p+1}(L)}{V(L)}
\end{aligned}
$$

gain/loss	probability	range	cause
$-\frac{C_i(L)}{A_i(L)}$	$(1-q)\frac{C_i(L)+iA_i(L)}{L(1-q)}$	$p \le i \le 2p-1$	insert
$-\frac{C_i(L)}{A_i(L)}$	$q\frac{iA_i(L)}{L(1-2q)}$	$p \le i \le 2p-1$	delete
$-\frac{C_i(L)}{A_i(L)}$	$q\frac{pA_p(L)}{L(1-2q)}\frac{A_i(L)}{V(L)}$	$p \le i \le 2p-1$	merge
$\frac{C_{i-1}(L)}{A_{i-1}(L)}$	$(1-q)\frac{C_{i-1}(L)+(i-1)A_{i-1}(L)}{L(1-q)}$	$p < i \le 2p-1$	insert
$\frac{C_{2p-1}(L)}{A_{2p-1}(L)}$	$(1-q)\frac{C_i(L)+iA_i(L)}{L(1-q)}$	$i = p$	insert
$\frac{C_{i+1}(L)}{A_{i+1}(L)}+1$	$q\frac{(i+1)A_{i+1}(L)}{L(1-2q)}$	$p \le i < 2p-1$	delete
$\frac{i}{2i-p}\frac{C_{2i-p}(L)}{A_{2i-p}(L)}$	$q\frac{pA_p(L)}{L(1-2q)}\frac{A_{2i-p}(L)}{V(L)}$	$i < p \le (3p-2)/2$	merge
$\frac{C_p(L)}{A_p(L)}+\frac{C_{2i-p+1}(L)}{A_{2i-p+1}(L)}+1$	$q\frac{pA_p(L)}{L(1-2q)}\frac{A_{2i-p+1}(L)}{V(L)}$	$i \le p \le (3p-2)/2$	merge
$\frac{C_p(L)}{A_p(L)}+\frac{i-p+1}{2i-p+2}\frac{C_{2i-p+2}(L)}{A_{2i-p+2}(L)}+1$	$q\frac{pA_p(L)}{L(1-2q)}\frac{A_{2i-p+2}(L)}{V(L)}$	$i \le p < (3p-2)/2$	merge
$\frac{C_p(L)}{A_p(L)}+1$	$q\frac{pA_p^2(L)}{L(1-2q)V(L)}$	$i = 2p-1$	merge

Table 19: Changes in values of C_i on an operation

$$+q\left(\frac{C_p(L)}{A_p(L)}+\frac{i+p-1}{2i-p+2}\frac{C_{2i-p+2}(L)}{A_{2i-p+2}(L)}+1\right)\frac{pA_p(L)}{L(1-2q)}\frac{A_{2i-p+2}(L)}{V(L)}$$

$$\begin{aligned}
&(L+1)qc_i(L+1) = \\
&\quad Lqc_i(L) - \frac{q}{1-2q}\frac{c_i(L)}{a_i(L)}(qia_i(L)+qc_i(L)+i(1-2q)a_i(L)+qp\frac{a_p(L)a_i(L)}{v(L)}) \\
&\quad +q(i+1)a_{i+1}(L)(\frac{qc_{i+1}(L)}{(1-2q)a_{i+1}(L)}+1)+\frac{qc_{i-1}(L)}{(1-2q)a_{i-1}(L)}(qc_{i-1}(L)+(i-1)(1-2q)a_{i-1}(L)) \\
&\quad +\frac{i}{2i-p}\frac{qc_{2i-p}(L)}{(1-2q)v(L)}qpa_p(L)+q(\frac{qc_p(L)pa_{2i-p+1}(L)}{(1-2q)v(L)}+\frac{qc_{2i-p+1}(L)pa_p(L)}{(1-2q)v(L)}+\frac{pA_pA_{2i-p+1}}{L(1-2q)V(L)}) \\
&\quad +q(\frac{qc_p(L)pa_{2i-p+2}(L)}{(1-2q)v(L)}+\frac{i-p+1}{2i-p+2}\frac{qpc_{2i-p+2}(L)a_p(L)}{(1-2q)v(L)}+\frac{pA_pA_{2i-p+2}}{L(1-2q)V(L)}))
\end{aligned}$$

$$\begin{aligned}
\Delta(L+1)qc_i(L+1) = &-c_i(L)(q+\frac{iq^2}{1-2q}+\frac{q^2}{1-2q}\frac{c_i(L)}{a_i(L)}+iq)+\frac{q^2(i+1)c_{i+1}(L)}{1-2q} \\
&+q(i+1)a_{i+1}(L)+c_{i-1}(L)(\frac{q^2c_{i-1}(L)}{(1-2q)a_{i-1}(L)}+(i-1)q) \\
&+\frac{q^2p}{(1-2q)v(L)}\Big[\frac{ic_{2i-p}(L)a_p(L)}{2i-p}+\Big(c_p(L)a_{2i-p+1}(L)+c_{2i-p+1}(L)a_p(L)+\frac{1-2q}{q}a_p(L)a_{2i-p+1}(L)\Big) \\
&+\Big(c_pa_{2i-p+2}(L)+\frac{(i-p+1)c_{2i-p+2}(L)a_p(L)}{2i-p+2}+\frac{1-2q}{q}a_p(L)a_{2i-p+2}(L)\Big)-a_p(L)c_i(L)\Big]
\end{aligned}$$

We can find the equilibrium point for $p < i < (3p-2)/2$ to get:

$$\begin{aligned}
0 = \quad &-c_i(q+\frac{iq^2}{1-2q}+\frac{q^2}{1-2q}\frac{c_i}{a_i}+iq)+\frac{q^2(i+1)c_{i+1}}{1-2q} \\
&+q(i+1)a_{i+1}+c_{i-1}(\frac{q^2c_{i-1}}{(1-2q)a_{i-1}}+(i-1)q) \\
&+\frac{q^2p}{(1-2q)v}\Big[\frac{ic_{2i-p}a_p}{2i-p}+\Big(c_pa_{2i-p+1}+c_{2i-p+1}a_p+\frac{1-2q}{q}a_pa_{2i-p+1}\Big) \\
&+\Big(c_pa_{2i-p+2}+\frac{(i-p+1)c_{2i-p+2}a_p}{2i-p+2}+\frac{1-2q}{q}a_pa_{2i-p+2}\Big)-a_pc_i\Big]
\end{aligned}$$

We can find the equilibrium point for the other cases in a similar manner.

References

[Baeza-Yates 89a] 'Expected Behavior of B^+-trees Under Random Insertions' R. Baeza-Yates to appear in Acta-Informatica Vol. 27 (1989)

[Baeza-Yates 89b] 'Performance of B^+ trees with Partial Expansions' R. Baeza-Yates, P. Larson *to appear in IEEE Trans. on Knowledge and Data Engineering Vol. 1* (1989)

[Bayer 77] Concurrency of Operations on B-trees' R. Bayer, M. Schkolnick *Acta Informatica 9* pp. 1-21 (1977)

[Eisenbarth 82] 'The Theory of Fringe Analysis and its Application to 2-3 Trees and B-trees' B. Eisenbarth, N. Ziviani, G. Gonnet, K. Mehlhorn, D. Wood, *Information and Control Vol. 55, No. 1* pp. 125-174 (1982)

[Driscoll 87] 'Modeling B-tree Insert Activity' J. Driscoll, S. Lang, L. Franklin *Information Processing Letters Vol. 26* pp. 5-18 (1987)

[Gupta 86] 'Approximate Storage Utilization of B-trees' G. Gupta, B. Srinivasan *Information Processing Letters 22*, pp 243-246 (1986)

[Knuth 73] D. Knuth 'The Art of Computer Programming, Vol. 3' *Addison Wesly* (1973)

[Kuspert 83] 'Storage Utilization in B*-trees with a Generalized Overflow Technique' *Acta Informatica Vol. 26, No. 4* pp. 35-56 (1983)

[Langenhop 85] 'An Efficient Model for Representing and Analyzing B-trees' *ACM-NCC* pp. 35-40 (1985)

[Leung 84] 'Approximate storage utilization of B-trees: A Simple Derivation and Generalizations' Clement Leung *Information Processing Letters 19*, pp 199-201 (1984)

[Mizoguchi 79] 'On the required Space for Random Split Trees' T. Mizoguchi *Allerton Conference, Monticello, Il* pp. 265-273 (1979)

[NAG] NAG (USA) Inc.; 1131 Warren Ave.; Downers Grove, Illinois 60515

[Nakamura 78] 'An Analysis of Storage Utilization Factor in Block Split Data Structuring Scheme' T. Nakamura, T. Mizoguchi *VLDB, Berlin* pp. 489-495 (1978)

[Quitzow 80] 'Space Utilization and Access Path Length in B-trees' K. Quitzow, M. Klopprogge *Information Systems Vol. 5* pp. 7-16 (1980)

[Ross 84] S. Ross 'A First Course in Probability' *Macmillan 1984*

[Wright 85] 'Some Average Performance Measures for the B-tree' William Wright *Acta Informatica 16*, (1985)

[Yao 78] 'On Random 2-3 Trees' Andrew Yao *Acta Informatica 9*, pp. 159-170 (1978)

[Zhang 87] 'Unsafe Operations in B-trees' Bin Zhang, Meichun Hsu *to appear in Acta Informatica*

p	utilization		probabability of splitting (merging)	
	analytical	simulation	analytical	simulation
5	43.02%	45.31%	$4.00 \cdot 10^{-2}$	$1.96 \cdot 10^{-2}$
10	39.34	39.48	$1.00 \cdot 10^{-2}$	$1.19 \cdot 10^{-3}$
15	38.21	38.40	$4.44 \cdot 10^{-3}$	$8.33 \cdot 10^{-5}$
20	37.66	39.77	$2.50 \cdot 10^{-3}$	$2.05 \cdot 10^{-5}$

Table 1: Merge-at-empty linear model and simulation for pure modifies: utilization and probabilities of splitting and merging

p	utilization		probability of splitting (merging)	
	analytical	simulation	analytical	simulation
5	47.51%	45.31%	$1.02 \cdot 10^{-2}$	$1.96 \cdot 10^{-2}$
10	42.67	39.48	$2.54 \cdot 10^{-4}$	$1.19 \cdot 10^{-3}$
15	40.87	38.40	$6.40 \cdot 10^{-6}$	$8.33 \cdot 10^{-5}$
20	39.93	39.77	$1.60 \cdot 10^{-7}$	$2.05 \cdot 10^{-5}$

Table 2: Merge-at-empty ghost model and simulation for pure modifies: utilization and probabilities of splitting and merging

p	utilization		probability of splitting		probability of deleting	
	analytical	simulation	analytical	simulation	analytical	simulation
5	57.81	54.29	$3.69 \cdot 10^{-2}$	$4.11 \cdot 10^{-2}$	$2.36 \cdot 10^{-3}$	$5.31 \cdot 10^{-3}$
10	62.38	58.34	$1.53 \cdot 10^{-2}$	$1.51 \cdot 10^{-2}$	$1.86 \cdot 10^{-6}$	$6.6 \cdot 10^{-5}$
15	64.36	60.86	$9.74 \cdot 10^{-3}$	$9.34 \cdot 10^{-3}$	$1.95 \cdot 10^{-9}$	0
20	65.36	62.39	$7.13 \cdot 10^{-3}$	$6.82 \cdot 10^{-3}$	$2.44 \cdot 10^{-12}$	0

Table 3: Merge-at-empty ghost model and simulation, $q = .45$ (10% more inserts than deletes)

p	utilization		probability of splitting		probability of deleting	
	analytical	simulation	analytical	simulation	analytical	simulation
5	54.88	50.31	$2.63 \cdot 10^{-2}$	$3.05 \cdot 10^{-2}$	$3.86 \cdot 10^{-3}$	$1.02 \cdot 10^{-2}$
10	60.40	53.54	$9.86 \cdot 10^{-3}$	$9.56 \cdot 10^{-3}$	$5.28 \cdot 10^{-6}$	$1.05 \cdot 10^{-4}$
15	63.36	57.17	$6.16 \cdot 10^{-3}$	$5.78 \cdot 10^{-3}$	$5.40 \cdot 10^{-9}$	0
20	64.82	58.94	$4.47 \cdot 10^{-3}$	$4.13 \cdot 10^{-3}$	$9.80 \cdot 10^{-12}$	0

Table 4: Merge-at-empty ghost model and simulation, $q = .47$ (6% more iinserts than deletes)

p		q as percentage						
		5	10	20	30	40	45	47
5	analytical	71.12	70.35	68.06	63.91	56.06	50.14	47.41
5	simulation	71.14	70.34	69.01	66.45	59.75	54.29	50.31
10	analytical	70.10	69.74	68.64	66.54	60.38	52.70	48.02
10	simulation	70.51	70.13	69.20	67.77	63.04	58.34	53.52
20	analytical	69.68	69.51	68.97	67.90	64.70	58.77	53.12
20	simulation	69.85	69.98	69.57	68.03	64.86	62.39	58.94
30	analytical	69.55	69.43	69.08	68.37	66.26	62.07	57.13
30	simulation	69.69	70.05	69.14	68.60	67.03	64.30	-
40	analytical	69.49	69.40	69.14	68.61	67.03	63.87	59.83
40	simulation	69.60	71.49	69.11	68.43	67.53	65.06	-
70	analytical	69.41	69.36	69.21	68.92	68.01	66.21	63.82
100	analytical	69.38	69.34	69.24	69.04	68.41	67.15	65.48

Table 5: Merge-at-empty linear model and simulation for varying node sizes ang percentages of deletes: utilization

p		q as percentage						
		5	10	20	30	40	45	47
5	ana.	$1.48 \cdot 10^{-1}$	$1.40 \cdot 10^{-1}$	$1.23 \cdot 10^{-1}$	$1.00 \cdot 10^{-1}$	$7.21 \cdot 10^{-2}$	$5.62 \cdot 10^{-2}$	$4.97 \cdot 10^{-2}$
5	sim.	$1.48 \cdot 10^{-1}$	$1.39 \cdot 10^{-1}$	$1.20 \cdot 10^{-1}$	$9.26 \cdot 10^{-2}$	$6.18 \cdot 10^{-2}$	$4.11 \cdot 10^{-2}$	$3.05 \cdot 10^{-2}$
10	ana.	$7.11 \cdot 10^{-2}$	$6.71 \cdot 10^{-2}$	$5.76 \cdot 10^{-2}$	$4.53 \cdot 10^{-2}$	$2.93 \cdot 10^{-2}$	$1.97 \cdot 10^{-2}$	$1.57 \cdot 10^{-2}$
10	sim.	$7.06 \cdot 10^{-2}$	$6.68 \cdot 10^{-2}$	$5.71 \cdot 10^{-2}$	$4.46 \cdot 10^{-2}$	$2.71 \cdot 10^{-2}$	$1.51 \cdot 10^{-2}$	$9.56 \cdot 10^{-3}$
20	ana.	$3.49 \cdot 10^{-2}$	$3.28 \cdot 10^{-2}$	$2.79 \cdot 10^{-2}$	$2.16 \cdot 10^{-2}$	$1.32 \cdot 10^{-2}$	$7.99 \cdot 10^{-3}$	$5.72 \cdot 10^{-3}$
20	sim.	$3.47 \cdot 10^{-2}$	$3.24 \cdot 10^{-2}$	$2.77 \cdot 10^{-2}$	$2.13 \cdot 10^{-2}$	$1.29 \cdot 10^{-2}$	$6.82 \cdot 10^{-3}$	$4.13 \cdot 10^{-3}$
30	ana.	$2.30 \cdot 10^{-2}$	$2.17 \cdot 10^{-2}$	$1.84 \cdot 10^{-2}$	$1.42 \cdot 10^{-2}$	$8.53 \cdot 10^{-3}$	$4.97 \cdot 10^{-3}$	$3.40 \cdot 10^{-3}$
30	sim.	$2.30 \cdot 10^{-2}$	$2.12 \cdot 10^{-2}$	$1.83 \cdot 10^{-2}$	$1.39 \cdot 10^{-2}$	$8.08 \cdot 10^{-3}$	$4.33 \cdot 10^{-3}$	-
40	ana.	$1.73 \cdot 10^{-2}$	$1.62 \cdot 10^{-2}$	$1.37 \cdot 10^{-2}$	$1.05 \cdot 10^{-2}$	$6.29 \cdot 10^{-3}$	$3.60 \cdot 10^{-3}$	$2.39 \cdot 10^{-3}$
40	sim.	$1.78 \cdot 10^{-2}$	$1.64 \cdot 10^{-2}$	$1.37 \cdot 10^{-2}$	$1.05 \cdot 10^{-2}$	$5.92 \cdot 10^{-3}$	$3.25 \cdot 10^{-3}$	-
70	ana.	$9.82 \cdot 10^{-3}$	$9.22 \cdot 10^{-3}$	$7.80 \cdot 10^{-3}$	$5.97 \cdot 10^{-3}$	$3.53 \cdot 10^{-3}$	$1.98 \cdot 10^{-3}$	$1.28 \cdot 10^{-3}$
100	ana.	$6.86 \cdot 10^{-3}$	$6.44 \cdot 10^{-3}$	$5.44 \cdot 10^{-3}$	$4.16 \cdot 10^{-3}$	$2.45 \cdot 10^{-3}$	$1.36 \cdot 10^{-3}$	$8.69 \cdot 10^{-4}$

Table 6: Merge-at-empty linear model and simulation for varying node sizes and percentages of deletes: probability of splitting

p		q as percentage						
		5	10	20	30	40	45	47
5	ana.	$3.93 \cdot 10^{-7}$	$7.75 \cdot 10^{-6}$	$1.95 \cdot 10^{-4}$	$1.63 \cdot 10^{-3}$	$9.04 \cdot 10^{-3}$	$1.95 \cdot 10^{-2}$	$2.62 \cdot 10^{-2}$
5	sim.	0	0	0	0	$2.10 \cdot 10^{-3}$	$5.31 \cdot 10^{-3}$	$1.02 \cdot 10^{-2}$
10	ana.	0	0	$5.04 \cdot 10^{-8}$	$6.27 \cdot 10^{-6}$	$3.17 \cdot 10^{-4}$	$1.89 \cdot 10^{-3}$	$3.74 \cdot 10^{-3}$
10	sim.	0	0	0	0	0	$6.60 \cdot 10^{-10}$	$1.05 \cdot 10^{-4}$
20	ana.	0	0	0	$3.44 \cdot 10^{-10}$	$1.45 \cdot 10^{-6}$	$6.74 \cdot 10^{-5}$	$2.98 \cdot 10^{-4}$
20	sim.	0	0	0	0	0	0	0
30	ana.	0	0	0	0	$1.15 \cdot 10^{-8}$	$4.16 \cdot 10^{-6}$	$4.12 \cdot 10^{-5}$
30	sim.	0	0	0	0	0	0	0
40	ana.	0	0	0	0	$1.14 \cdot 10^{-10}$	$3.21 \cdot 10^{-7}$	$7.14 \cdot 10^{-6}$
40	sim.	0	0	0	0	0	0	0
70	ana.	0	0	0	0	0	$2.65 \cdot 10^{-10}$	$6.63 \cdot 10^{-8}$
100	ana.	0	0	0	0	0	0	$9.07 \cdot 10^{-10}$

Table 7: Merge-at-empty linear model and simulation for varying node sizes and percentages of deletes: probability of merging

		Utilization						
p		$q = .1$	$q = .2$	$q = .3$	$q = .4$	$q = .45$	$q = .47$	$q = .5$
20	ana.	70.09	70.36	70.48	69.70	67.92	66.43	62.91
20	sim.	70.09	70.31	70.34	70.24	69.24	68.33	66.07
30	ana.	69.98	70.37	70.72	70.41	68.93	67.37	61.43
30	sim.	70.41	70.10	70.54	69.98	69.85	68.98	64.47
40	ana.	69.94	70.38	70.86	70.86	69.66	68.22	60.51
40	sim.	70.05	70.49	70.49	70.63	69.59	69.54	63.30

Table 8: Comparison analytical and simulation space utilization for merge-at-half B-trees

		Pr[split on insert]						
p		$q = .1$	$q = .2$	$q = .3$	$q = .4$	$q = .45$	$q = .47$	$q = .5$
20	ana.	.0331	.0284	.0221	.0136	.00852	.00669	.00711
20	sim.	.0340	.0321	.0279	.0201	.0144	.0132	.0118
30	ana.	.0218	.0185	.0142	.00843	.00483	.00329	.00226
30	sim.	.0225	.0211	.0178	.0128	.00584	.00491	.00521
40	ana.	.0162	.0138	.0105	.00615	.00344	.00224	.00126
40	sim.	.0170	.0153	.0133	.00939	.00443	.00331	.00226

Table 9: Comparison analytical and simulation probability of splitting on an insert for merge-at-half B-trees

p		Pr[merge on delete]						
		$q = .1$	$q = .2$	$q = .3$	$q = .4$	$q = .45$	$q = .47$	$q = .5$
20	ana.	.0644	.0571	.0478	.0384	.0391	.0450	.0762
20	sim.	.0691	.0645	.0580	.0494	.0446	.0483	.0572
30	ana.	.0428	.0372	.0299	.0210	.0181	.0193	.0465
30	sim.	.0452	.0432	.0379	.0305	.0193	.0163	.0330
40	ana.	.0320	.0276	.0217	.0143	.0109	.0107	.0324
40	sim.	.0318	.0320	.0283	.0215	.0122	.0110	.0193

Table 10: Comparison analytical and simulation probability of merging on an delete for merge-at-half B-trees

p		Pr[restructure on operation]						
		$q = .1$	$q = .2$	$q = .3$	$q = .4$	$q = .45$	$q = .47$	$q = .5$
20	half	.0362	.0341	.0298	.0235	.0223	.0247	.0417
20	empty	.0291	.0221	.0149	.00774	.00378	.00223	.00005
30	half	.0261	.0224	.0189	.0135	.0108	.0108	.0244
30	empty	.0190	.0146	.00973	.00513	.00283	.00180	.00000
40	half	.0178	.0166	.0139	.00941	.00680	.00622	.0168
40	empty	.0147	.0109	.00735	.00355	.00178	.00125	.00000

Table 11: Comparison of merge-at-half and merge-at-empty probability of restructuring in an operation $(qP_m + (1-q)P_s)$

	analysis	simulation				
		$q=0$	$q=.1$	$q=.2$	$q=.3$	$q=.4$
a_{10}	.0136	.0134	.0144	.0131	.0128	.0130
a_{11}	.0113	.0104	.00972	.0106	.0105	.0110
a_{12}	.00959	.00934	.00834	.00910	.00948	.00930
a_{13}	.00822	.00767	.00739	.00851	.00787	.00853
a_{14}	.00712	.00723	.00691	.00707	.00754	.00630
a_{15}	.00623	.00622	.00606	.00692	.00633	.00649
a_{16}	.00550	.00576	.00655	.00560	.00584	.00535
a_{17}	.00489	.00550	.00509	.00465	.00553	.00533
a_{18}	.00437	.00434	.00490	.00484	.00434	.00440
a_{19}	.00393	.00432	.00454	.00388	.00388	.00448

Table 12: Comparison analytical vs. simulation values of $a_{2,i}$ for various q (distribution of second level nodes, bottom-up algorithm)

	$q=0$		$q=.1$		q=.2		q=.3		q=.4	
	ana	sim	ana	sim	ana	sim	ana	sim	ana	sim
a_9	.0078	.0071	.0078	.0068	.0078	.0069	.0079	.0072	.0079	.0070
a_{10}	.0136	.0118	.0136	.0121	.0136	.0135	.0136	.0134	.0136	.0127
a_{11}	.0113	.0116	.0113	.0119	.0113	.0108	.0113	.0109	.0114	.0112
a_{12}	.0096	.0097	.0096	.0092	.0096	.0091	.0096	.0097	.0096	.0094
a_{13}	.0082	.0083	.0082	.0084	.0082	.0085	.0082	.0080	.0082	.0089
a_{14}	.0071	.0074	.0071	.0070	.0071	.0072	.0071	.0073	.0071	.0070
a_{15}	.0062	.0063	.0062	.0070	.0062	.0061	.0063	.0064	.0062	.0064
a_{16}	.0055	.0055	.0055	.0057	.0055	.0060	.0055	.0055	.0055	.0059
a_{17}	.0049	.0050	.0049	.0053	.0049	.0052	.0049	.0052	.0049	.0047
a_{18}	.0044	.0049	.0044	.0043	.0044	.0046	.0044	.0045	.0044	.0047
a_{19}	.0003	.0002	.0003	.0002	.0002	.0003	.0002	.0002	.0001	.0002

Table 13: Comparison analytical vs. simulation values of $a_{2,i}$ for various q (distribution of second level nodes, top-down algorithm)

	utilization		
h	$q=0$	$q=.2$	$q=.4$
2	68.473	68.470	68.465
3	68.460	68.460	68.460
4	68.460	68.460	68.460

Table 14: Space utilization of various levels of a top-down tree, $p = 30$

h	E'_h	V'_h	σ'_h
1	13.4	7.73	2.78
2	179	1486	38.5
3	2394	2.67×10^5	517
4	32019	4.79×10^7	6919
5	4.28×10^5	8.56×10^9	92543

Table 15: Variance of distribution of number of items covered in a bottom-up B-tree, $p = 10$

		bottom up			top-down		
h	i	$E_r(h,i)$	h_{sim}	i_{sim}	$E_r(h,i)$	h_{sim}	i_{sim}
4	12	27988	4	12	25085	4	11
4	13	30320	4	15	27176	4	12
4	14	32652	4	16	29266	4	14
4	15	34984	4	16	31357	4	15
4	16	37316	4	16	33448	4	18
4	17	39648	4	16	35538	4	18
4	18	41980	4	16	37629	4	18
4	19	44312	4	18	-	-	-
5	2	46644	4	18	39719	4	18
5	2	62387	5	2	52918	5	2
5	3	93580	5	3	79375	5	3
5	4	124774	5	4	105835	5	4
5	5	155867	5	4	132294	5	4
5	6	187161	5	7	158752	5	6
5	7	218354	5	8	185211	5	8
5	8	249548	5	8	211670	5	8
5	9	280742	5	9	238129	5	8
5	10	311396	5	9	264588	5	8
5	11	343130	5	10	291047	5	11
5	12	374324	5	14	317505	5	12

Table 16: Comparison of predicted and simulated root height and order for various $E_r(h,i)$

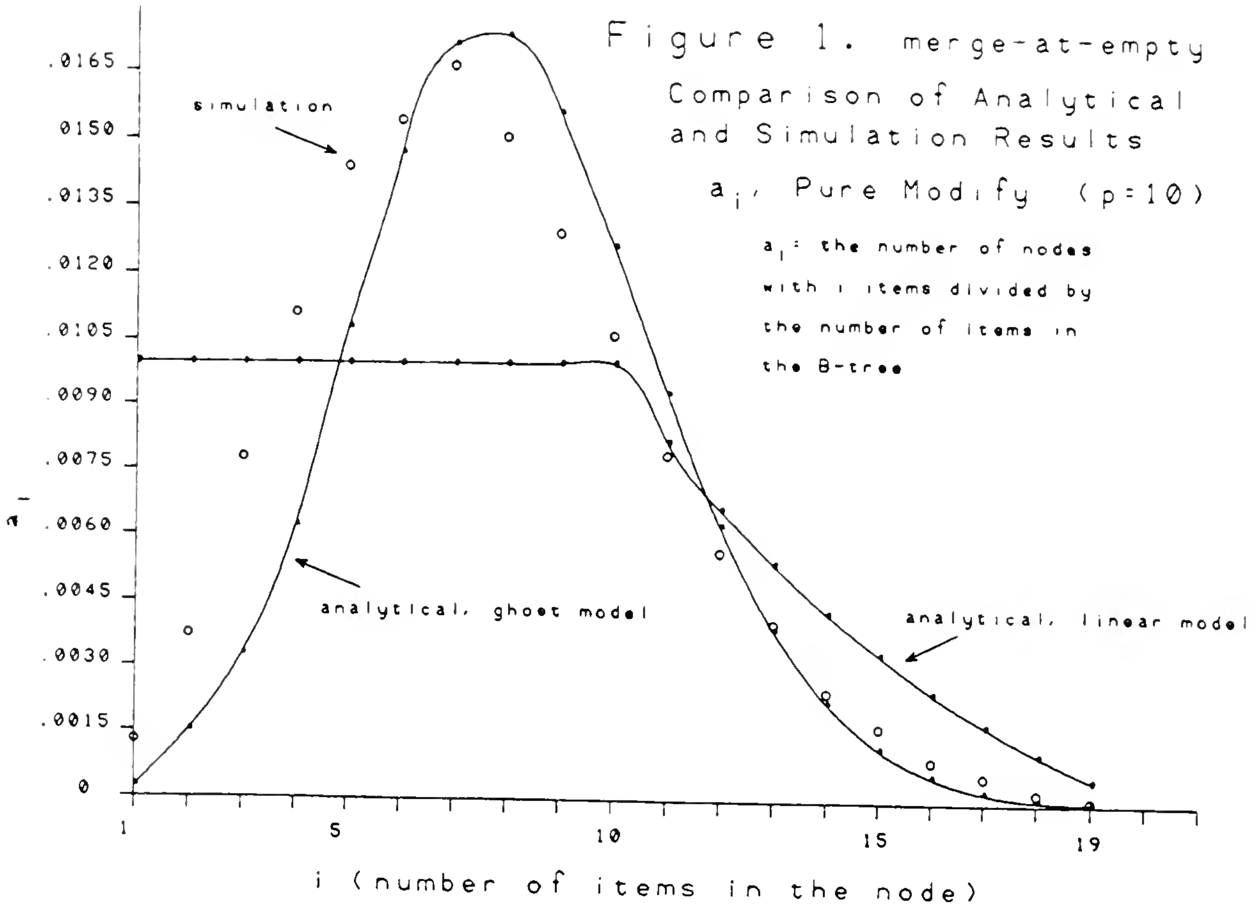


Figure 1. merge-at-empty Comparison of Analytical and Simulation Results a_i, Pure Modify (p=10)

Figure 2.

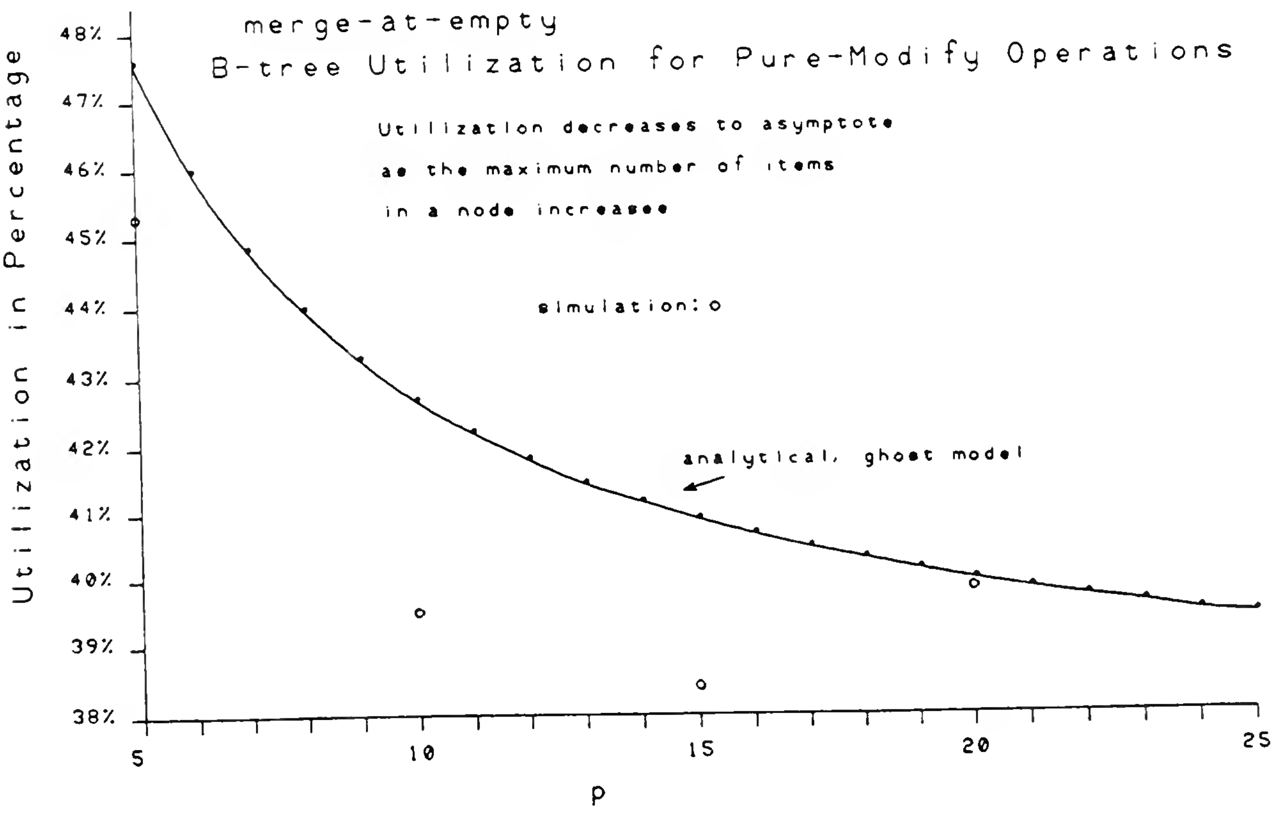

merge-at-empty B-tree Utilization for Pure-Modify Operations

Figure 3.

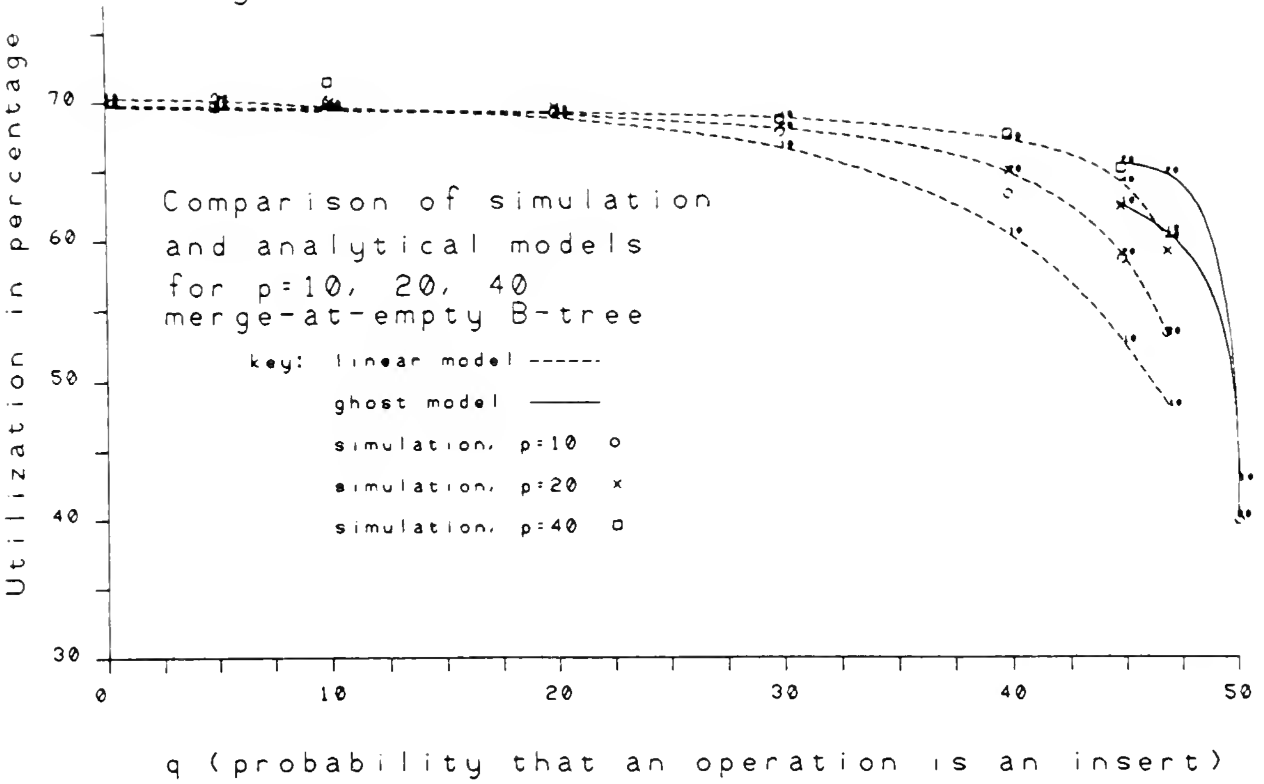
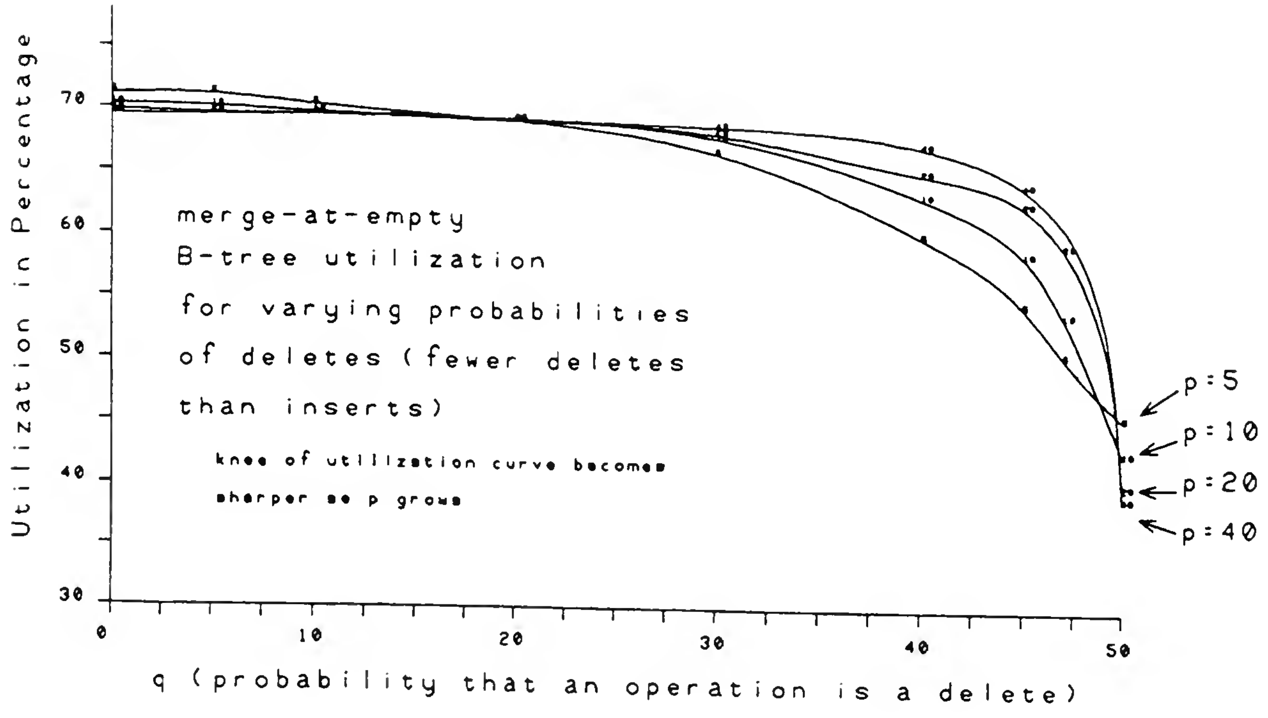


Figure 4.



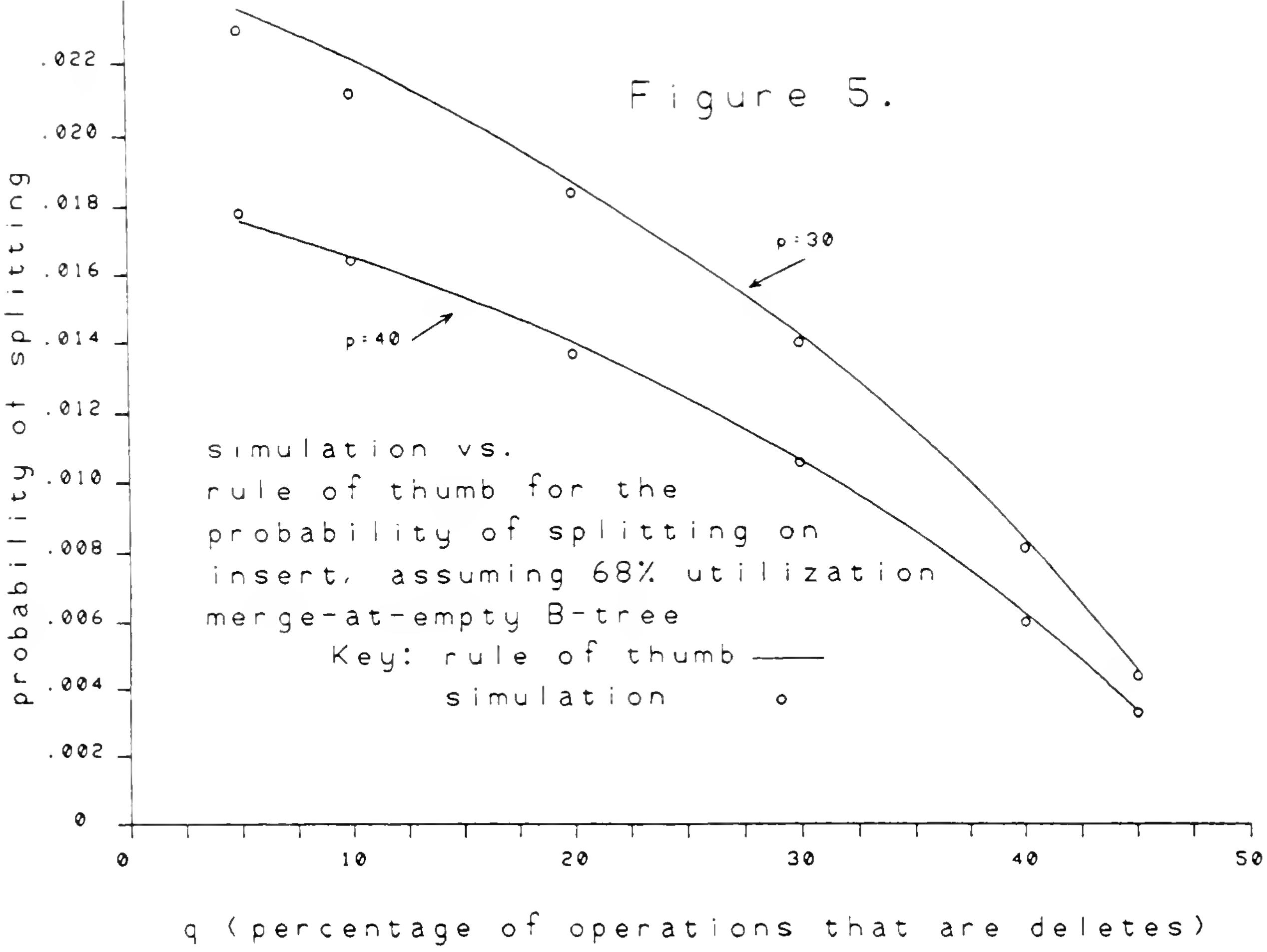


Figure 5.

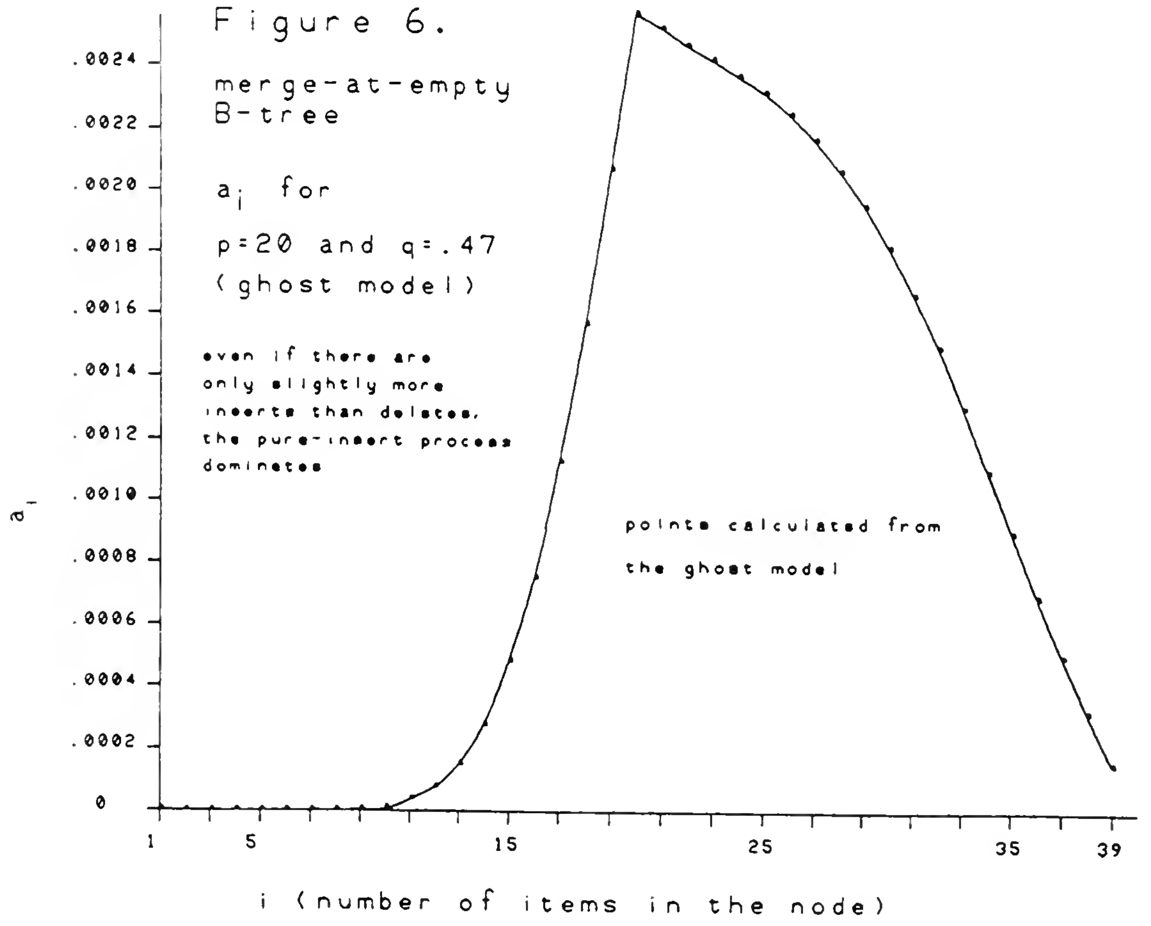


Figure 6.

Figure 7.

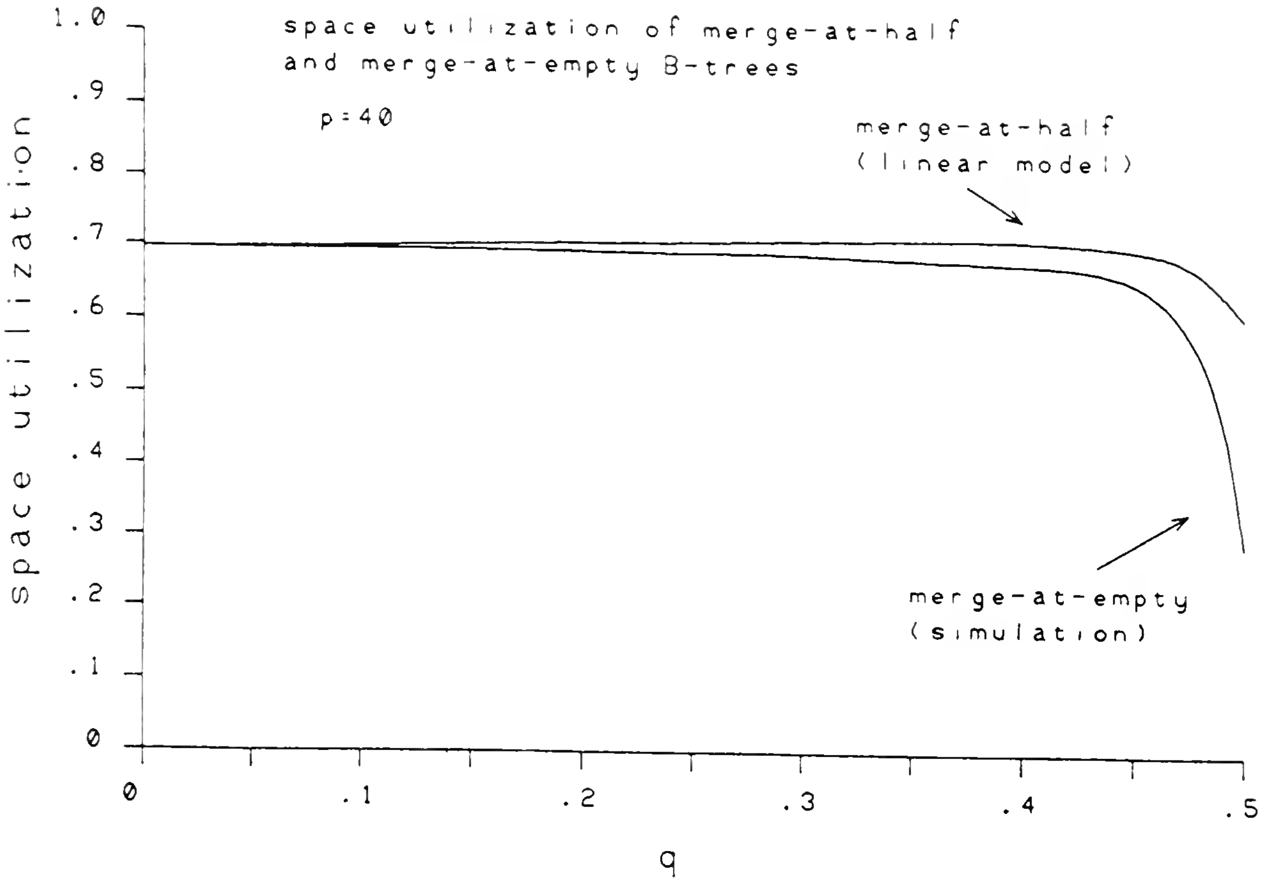
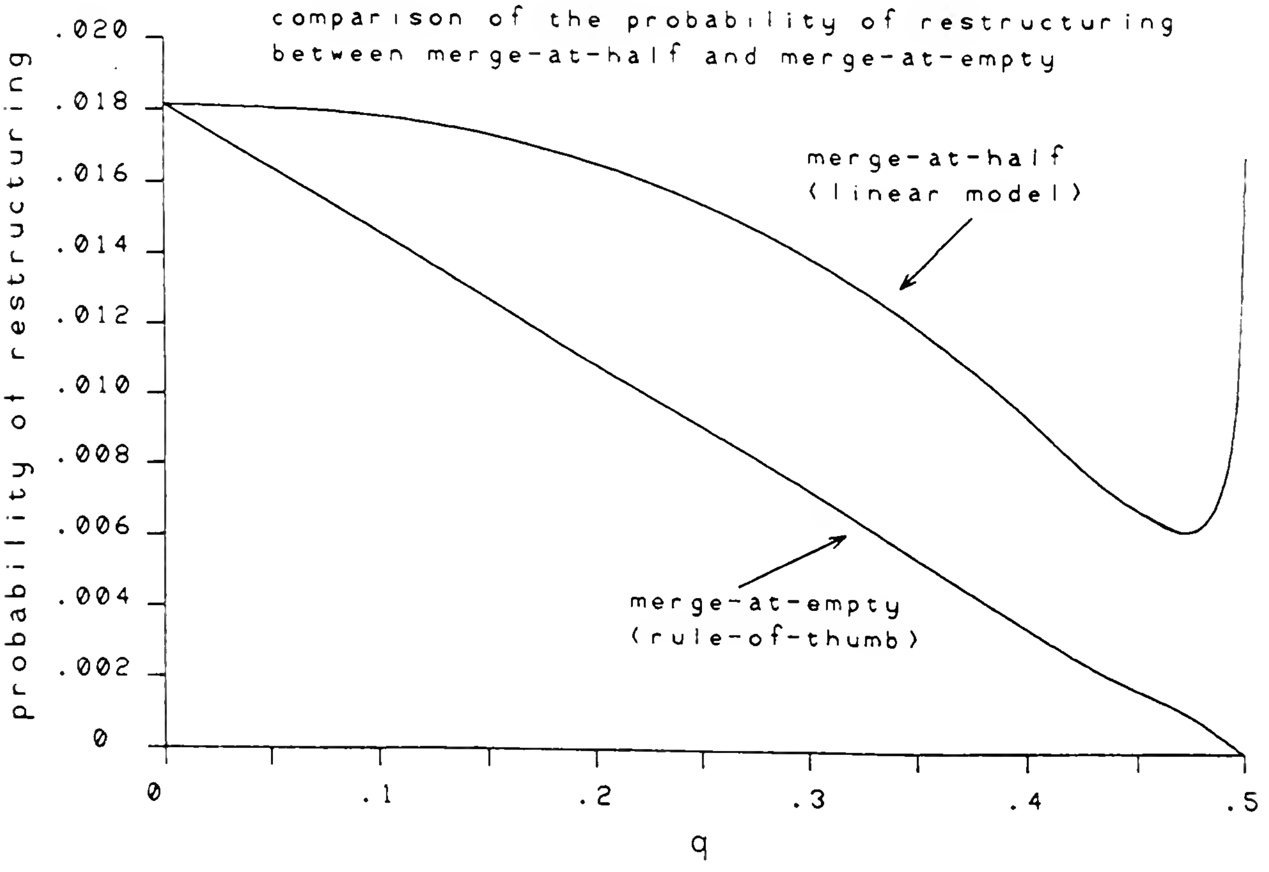


Figure 8.



NYU COMPSCI TR-453
Johnson, Theodore
Random B-trees with inserts
and deletes.
c.2

NYU COMPSCI TR-453
Johnson, Theodore
Random B-trees with inserts
and deletes.
c.2